Thèse de Doctorat

INSA Lyon, membre de l'Université de Lyon

École Doctorale N°512 — Informatique et Mathématiques

Discipline : Informatique

N° d'ordre NNT : XXX

Improving the Accuracy of Printed Mathematical Expression Recognition for Patent Publication

---

Soutenue publiquement le 15/12/2026, par :

François WIECKOWIAK

Devant le jury composé de :

Nom	Grade	Institution	Rôle
Harold MOUCHÈRE	Professeur des Universités	Université de Nantes	Rapporteur
Mickaël COUSTATY	Maître de Conférences	Université de La Rochelle	Rapporteur
Bertrand COÜASNON	Professeur des Universités	Université de Rennes	Examinateur
Véronique EGLIN	Professeure des Universités	INSA Lyon	Directrice de thèse
Tony BONNET	Encadrant Industriel	Luminess	Directeur de thèse
Laëtitia ROUSSEAU	Ingénieure IA	Luminess	Invitée
Stéphane BRES	Maître de Conférences	INSA Lyon	Invité

Département FEDORA – INSA Lyon - Ecoles Doctorales

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
ED 206 CHIMIE	**CHIMIE DE LYON** https://edchimie.universite-lyon.fr Sec. : Renée EL MELHEM Bâtiment Blaise Pascal, UCB Lyon 1 Tél : 04.72.43.80.46 secretariat@edchimie-lyon.fr	**M. Bruno ANDRIOLETTI** Université Claude Bernard Lyon 1 Bâtiment F308, BP 2077 43 Boulevard du 11 novembre 1918 69616 Villeurbanne directeur@edchimie-lyon.fr
ED 341 E2M2	**ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION** https://e2m2.universite-lyon.fr Sec. : Bénédicte LANZA Bâtiment Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 secretariat.e2m2@univ-lyon1.fr	**M. Marc LEMAIRE** Université Claude Bernard Lyon 1 Laboratoire MAP – Bâtiment LWOFF 10 rue Dubois 69622 Villeurbanne CEDEX e2m2.codir@listes.univ-lyon1.fr
ED 205 EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTÉ** https://ediss.universite-lyon.fr Sec. : Bénédicte LANZA Bâtiment Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 secretariat.ediss@univ-lyon1.fr	**M. Jérôme LAMARTINE** LBTI - UMR5305 CNRS – Université Claude Bernard Lyon 1 Equipe Intégrité Fonctionnelle du Tissu Cutané IBCP - 7 passage du Vercors 69367 Lyon Cedex 07 Tél : 04.72.72.26.66 jerome.lamartine@univ-lyon1.fr
ED 34 EDML	**MATÉRIAUX DE LYON** https://ed34.universite-lyon.fr Sec. : Yann DE ORDENANA Tél : 04.72.18.62.51 yann.de-ordenana@ec-lyon.fr	**M. Stéphane BENAYOUN** Ecole Centrale de Lyon Laboratoire LTDS – 36 avenue Guy de Collongue 69134 Ecully CEDEX Tél : 04.72.18.64.37 stephane.benayoun@ec-lyon.fr
ED 160 EEA	**ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE** https://edeea.universite-lyon.fr Sec. : Philomène TRECOURT Bâtiment Charlotte Perriand, INSA Lyon Tél : 04.72.43.71.70 secretariat.edeea@insa-lyon.fr	**M. Philippe DELACHARTRE** INSA LYON Laboratoire CREATIS Bâtiment Blaise Pascal, 7 avenue Jean Capelle 69621 Villeurbanne CEDEX Tél : 04.72.43.88.63 philippe.delachartre@insa-lyon.fr
ED 512 INFOMATHS	**INFORMATIQUE ET MATHÉMATIQUES** http://edinfomaths.universite-lyon.fr Sec. : Renée EL MELHEM Bâtiment Blaise Pascal, UCB Lyon 1 Tél : 04.72.43.80.46 infomaths@univ-lyon1.fr	**M. Dragos IFTIMIE** Université Claude Bernard Lyon 1 Bâtiment Braconnier, Bureau 232 21 Avenue Claude Bernard 69622 Villeurbanne Cedex Tél : 04.72.44.79.58 direction.infomaths@listes.univ-lyon1.fr
ED 162 MEGA	**MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE** https://edmega.universite-lyon.fr Sec. : Philomène TRECOURT Bâtiment Charlotte Perriand, INSA Lyon Tél : 04.72.43.71.70 mega@insa-lyon.fr	**M. Etienne PARIZET** INSA Lyon Laboratoire LVA Bâtiment St. Exupéry, 25 bis avenue Jean Capelle 69621 Villeurbanne CEDEX etienne.parizet@insa-lyon.fr
ED 483 ScSo	**ScSo**[1] – https://edsciencessociales.universite-lyon.fr Sec. : Mélina FAVETON Tél : 04.78.69.77.79 melina.faveton@univ-lyon2.fr	**M. Bruno MILLY** Université Lyon 2 Campus Berges du Rhône 18, quai Claude Bernard Bureau BEL 204 69365 LYON CEDEX 07 bruno.milly@univ-lyon2.fr

[1] ScSo : Architecture, Ergonomie, Géographie-Aménagement-Urbanisme, Histoire-Histoire de l'art, Mondes anciens, Science politique, Sociologie-Anthropologie

Remerciements

Quelle aventure ! Je ne pensais pas en arriver là, le 16 décembre 2021, lorsque j'ai envoyé un mail à Tony Bonnet pour lui demander : « Est-ce que Jouve [Aujourd'hui, Luminess] propose des sujets de recherche en thèse CIFRE ? » Et pourtant, j'arrive doucement à la fin de cette belle aventure, dont je ressors humble, satisfait et impatient de poursuivre ma route dans ce beau domaine qu'est la recherche en vision par ordinateur.

Merci aux membres du jury pour votre temps, pour avoir accepté de relire cette thèse et de participer à ma soutenance. J'espère que vous en apprendrez beaucoup, à travers ce manuscrit, sur ce formidable sujet qu'est la publication des brevets et la reconnaissance d'expressions mathématiques.

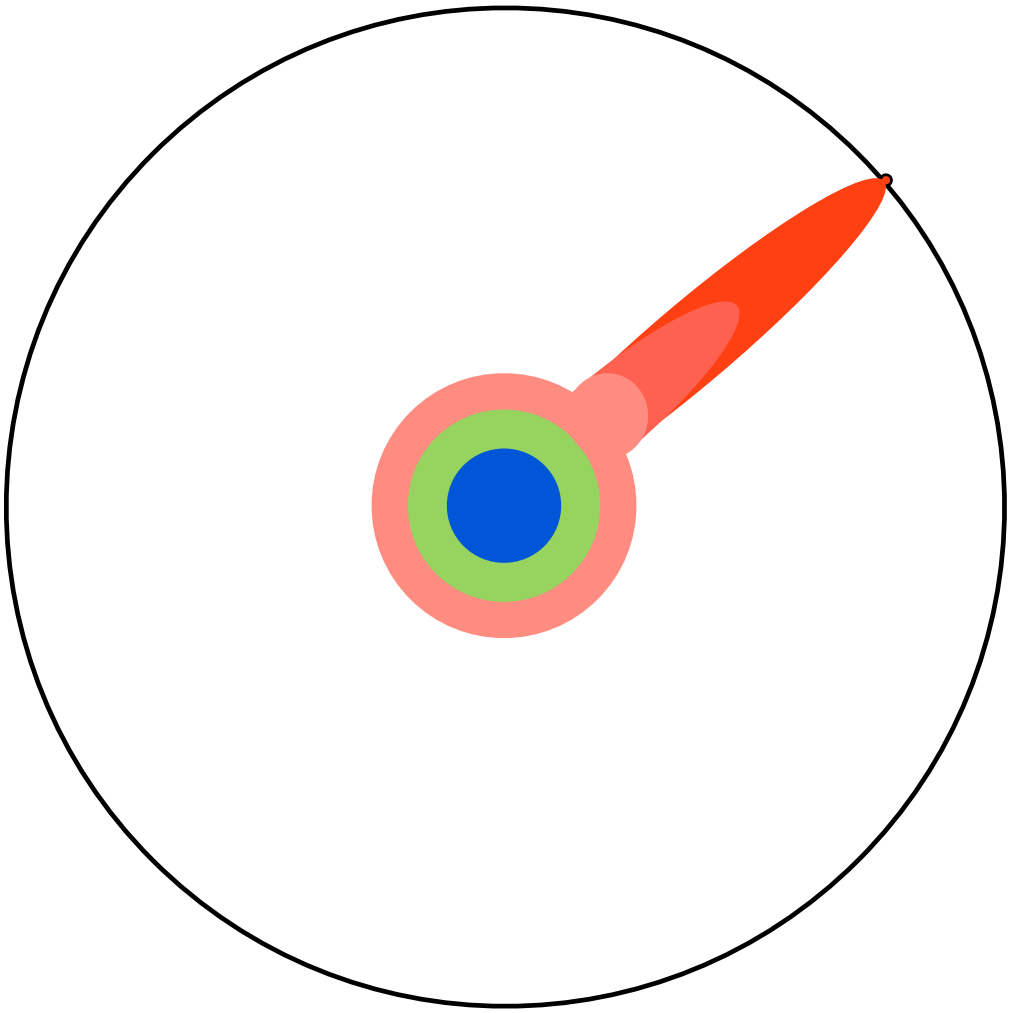
Je tiens à remercier très sincèrement mon équipe encadrante, et en particulier ma directrice et mon directeur de thèse, Véronique Eglin et Tony Bonnet, pour leur disponibilité sans faille, leurs encouragements et leur implication forte dans la préparation des deux beaux articles issus de cette thèse. Je remercie également très chaleureusement Laëtitia Rousseau et Stéphane Bres pour leurs remarques et leurs idées toujours aussi pertinentes. Les points hebdomadaires avec mon équipe et cette effervescence intellectuelle vont beaucoup me manquer.

Je remercie toutes les personnes que j'ai rencontrées chez Luminess. Par toutes vos réponses à mes questions et les discussions que nous avons eues, vous avez contribué, d'une manière ou d'une autre, à l'aboutissement de ce manuscrit. En particulier, merci à Antoine et Samuel pour tout ce temps passé à m'écouter réfléchir à voix haute au bureau, ainsi que pour toutes nos discussions sur nos travaux respectifs. Je vous souhaite le meilleur dans vos aventures professionnelles et cyclistes respectives, sans accidents et sans blessures s'il vous plaît.

Je remercie ma compagne, Séphora, avec qui je partage ma vie depuis déjà neuf années et qui a toujours été présente dans les bons comme dans les moins bons moments. Merci d'être là au quotidien, de supporter mes humeurs, mes deadlines et mes moments de doute, et de m'aider à m'ouvrir davantage aux autres et à moi-même. Je t'aime.

Je remercie ma famille, sans qui la personne que je suis aujourd'hui serait très différente. Merci de m'avoir toujours soutenu dans mes études, d'avoir été derrière moi et de m'avoir poussé à aller aussi loin dans mes études. Merci d'avoir cru en moi.

Enfin, merci à toutes les personnes que je n'ai pas explicitement citées ici : les membres du GRCE, de « La Star Maison », « The Mountain », « Le Cellier », « L'Élite », « DAPAC », « It's Warding Time », et bien d'autres. Je suis convaincu que chaque être humain est façonné par les personnes qui l'entourent. Si j'ai eu la chance de vous côtoyer un jour, alors il y a un petit morceau de vous dans cette thèse, et je vous remercie pour votre contribution.



The Illustrated Guide to a Ph.D., créé par Matt Might. Reproduit sous licence Creative Commons Attribution-NonCommercial 2.5. Source : http://matt.might.net/articles/phd-school-in-pictures/.

Résumé

Mots-clés : *Reconnaissance d'Expressions Mathématiques, Reconnaissance Optique de Caractères, Évaluation Multimodale, Vérification Post-OCR, Apprentissage de Métriques, Demandes de Brevet.*

Pour permettre la publication des brevets auprès de l'Office européen des brevets, l'entreprise Luminess est chargée de normaliser des demandes de brevet reçues dans des formats très hétérogènes en documents XML et PDF standardisés. Ces documents sont ensuite rendus accessibles à l'ensemble de la communauté scientifique et peuvent être facilement recherchés et retrouvés grâce à une annotation précise et normalisée de leur contenu. Cependant, des erreurs de reconnaissance peuvent altérer ces annotations et empêcher la recherche ou la récupération de documents pertinents.

Cette thèse porte sur l'automatisation de la reconnaissance des expressions mathématiques dans les demandes de brevet. Ces expressions, particulièrement complexes à reconnaître automatiquement, sont actuellement transcrites et vérifiées manuellement à plusieurs reprises afin de satisfaire aux exigences de qualité de l'Office européen des brevets. L'objectif de cette thèse est ainsi de réduire la charge associée à leur transcription et à leur vérification, tout en maintenant un niveau de qualité supérieur ou égal à 99,99 %.

Une première contribution consiste en un protocole d'évaluation multimodale permettant d'analyser les performances des systèmes de reconnaissance optique de caractères selon différentes représentations des expressions mathématiques. Cette évaluation met en évidence les limites des approches existantes sur les documents de brevet et conduit à la création de PatentME-600k, un jeu de données de plus de 600 000 expressions mathématiques extraites de brevets. Sur cette base, MML-Net, une architecture Vision Transformer encodeur-décodeur, est proposée pour la reconnaissance directe des expressions mathématiques en MathML. Le modèle atteint un score de similarité de Levenshtein de 93,7 % et dépasse les performances de l'état de l'art sur cette tâche.

Afin de permettre l'intégration de MML-Net dans une chaîne de production industrielle, plusieurs briques complémentaires sont développées. Un ordonnanceur basé sur un Random Forest et des caractéristiques visuelles sélectionne les expressions pour

lesquelles le modèle est susceptible de produire une transcription correcte, afin de réserver l'intervention humaine aux cas les plus complexes. Un vérificateur post-OCR, basé sur une architecture siamoise et entraîné spécifiquement sur les erreurs de MML-Net, détecte ensuite automatiquement les prédictions susceptibles d'être incorrectes et les oriente vers une validation humaine. Enfin, une contribution porte sur l'assistance à la saisie et la réduction des erreurs introduites lors des opérations manuelles. Différentes stratégies d'intégration de ces briques dans la chaîne de production de Luminess sont étudiées afin de combiner reconnaissance automatique et validation humaine, d'augmenter le niveau d'automatisation et de préserver les exigences de qualité du processus industriel.

Abstract

Keywords: *Mathematical Expression Recognition, Optical Character Recognition, Multimodal Evaluation, Post-OCR Verification, Metric Learning, Patent Applications.*

To allow the publication of patents by the European Patent Office, Luminess is responsible for standardizing patent applications received in highly heterogeneous formats into standardized XML and PDF documents. These documents are then made accessible to the scientific community and can be easily searched and retrieved thanks to precise and standardized annotation of their content. However, recognition errors can affect these annotations and prevent relevant documents from being found or retrieved.

This thesis focuses on the automation of mathematical expression recognition in patent applications. These expressions are particularly challenging to recognize automatically and are currently transcribed and manually verified several times to meet the quality requirements of the European Patent Office. The objective of this thesis is thus to reduce the workload associated with their transcription and verification while maintaining a quality level of at least 99.99%.

The first contribution consists of a multimodal evaluation framework for analyzing the performance of optical character recognition systems according to different representations of mathematical expressions. This evaluation highlights the limitations of existing approaches on patent documents and leads to the creation of PatentME-600k, a dataset containing more than 600,000 mathematical expressions extracted from patents. Using this dataset, MML-Net, a Vision Transformer encoder-decoder architecture, is proposed for the recognition of mathematical expressions in MathML. The model achieves a Levenshtein similarity score of 93.7% and outperforms state-of-the-art methods on this task.

To enable the integration of MML-Net into an industrial production pipeline, several complementary components are developed. A scheduler based on a Random Forest and visual features selects the expressions for which the model is most likely to produce a correct transcription, allowing human intervention to be reserved for more complex cases. A post-OCR verifier, based on a Siamese architecture and specifically trained on MML-Net errors, then automatically detects potentially incorrect predictions and routes

them for human validation. Finally, a final contribution focuses on assisting operators during transcription and reducing errors introduced during manual operations. Different strategies for integrating these components into the Luminess production pipeline are investigated in order to combine automatic recognition and human validation, increase the level of automation, and preserve the quality requirements of the industrial process.

Table of Contents

List of Figures

List of Tables

Chapter 1

General Introduction

Figure 1.1: Illustration of Jorge Luis Borges' *Library of Babel*, or *Infinite Library*, from Erik Desmazières. Its $1.956 \times 10^{1834097}$ 410-page books contain every possible permutation of letters. This means that every book that has been written, could be written in the future, and all the meaningless strings of letters are in the library. The hard part is finding the relevant book one is looking for.

1.1 Introduction

> *The universe (which others call the Library) is composed of an indefinite, perhaps infinite number of hexagonal galleries... The arrangement of the galleries is always the same: Twenty bookshelves, five to each side, line four of the hexagon's six sides... each bookshelf holds thirty-two books identical in format; each book contains four hundred ten pages; each page, forty lines; each line, approximately eighty black letters*
>
> — *Jorge Luis Borges, The Library of Babel, 1941*

Jorge Luis Borges' *Library of Babel*, or *Infinite Library* (Figure 1.1), is filled with every possible book that could ever be written, containing every possible combination of letters, spaces, and punctuation marks. Almost all of these books consist of meaningless gibberish, random sequences of characters that rarely form recognizable words in any language. However, one may be fortunate enough to find a book that describes, in great detail, everything good that will happen the next day. Sadly, there also exists a colossal number of books detailing every possible way in which that same day could go wrong. One could even find a book containing the exact text of this introduction[1].

With an alphabet of originally 25 symbols, books of 410 pages, each containing 40 lines of 80 characters, the Library is estimated to contain $25^{410\times40\times80} \approx 1.956 \times 10^{1834097}$ books (referred to as the Borges Number). The calculation of the time required to read every one of these books is left as an exercise for the reader.

The Library of Babel illustrates the challenge of information retrieval: when the volume of information becomes enormous, the central problem is no longer only storage, but finding the relevant information. Modern information retrieval systems can easily locate relevant documents and extract the desired information. However, they can only operate on documents that have been converted into machine-readable format. Text printed on paper must first be scanned into raster images then converted into text through Optical Character Recognition (OCR).

OCR for document recognition has a long development history. One of the earliest systems was developed by Emanuel Goldberg in the 1920s [1]. His *Statistical Machine* used light and punched cards to identify documents that had previously been recorded on microfilm. Early OCR systems then gradually became able to recognize printed characters, such as in [2] in 1961, which used autocorrelation to identify individual characters from their visual patterns. Later, OCR systems became capable of recognizing complete documents, such as ABBYY FineReader 1.0, released in 1993, which could recognize full pages of printed text [2].

[1] Volume 5 of shelf 5, wall 1 of hexagon ID 0qnz26shtvcu5nc... See https://libraryofbabel.info/bookmark.cgi?jth.hpwoyobvvwlf.izd214, accessed 2026-08-19.

[2] https://pdf.abbyy.com/blog/finereader-25-years/, accessed 2026-08-17

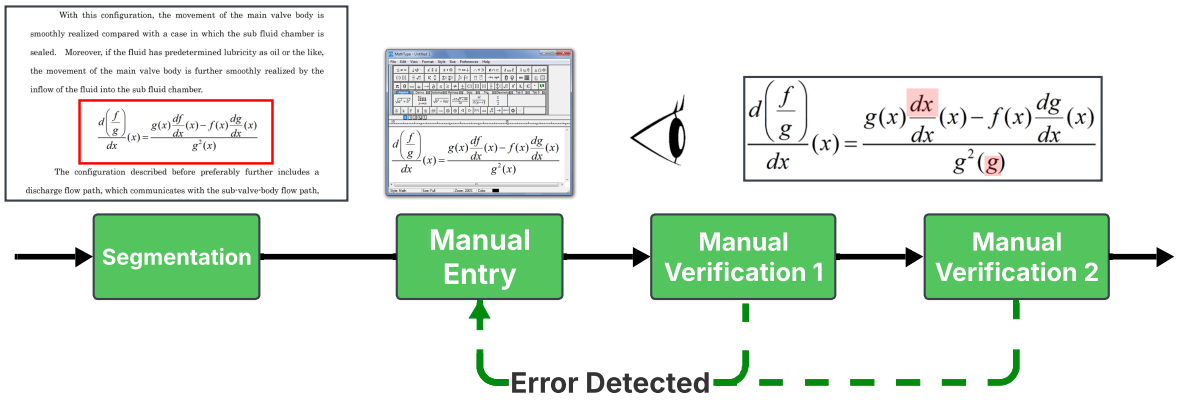
As OCR systems evolved from recognizing isolated characters to processing whole documents, the role of document layout became increasingly important. In a full-page document, the arrangement of visual elements provides important information about their relationships. This is particularly the case in technical and scientific documents. Unlike the books in the Library of Babel, these documents are not limited to a small set of symbols. They can contain a large variety of characters and visual elements arranged in complex ways, such as tables, diagrams, chemical formulas, and mathematical expressions. Among these elements, mathematical expressions represent a particularly challenging case. They contain a large number of symbols specific to mathematics and exhibit a high degree of two-dimensional structural complexity due to the presence of subscripts, superscripts, fractions, roots, matrices, and nested expressions. Recognizing such expressions requires more than simply identifying individual characters. The structure and relationships between the different elements must also be understood. This makes mathematical expressions a particularly challenging case for OCR and motivates the development of specialized mathematical OCR systems capable of reconstructing a structured representation from a two-dimensional visual arrangement.

In this work, we study OCR systems specialized in mathematical expression recognition, which consists in converting a visual representation of mathematical content into markup language, typically LaTeX. The work addresses offline recognition from static images, as opposed to online recognition which relies on stroke data captured by input devices. While approaches developed for online handwritten expression recognition offer relevant methodological insights, this thesis primarily treats offline printed expressions, as these two problems involve distinct data, models, and challenges. For interested readers, an overview of handwritten mathematical expression recognition is provided in Chapter 1 of [3].

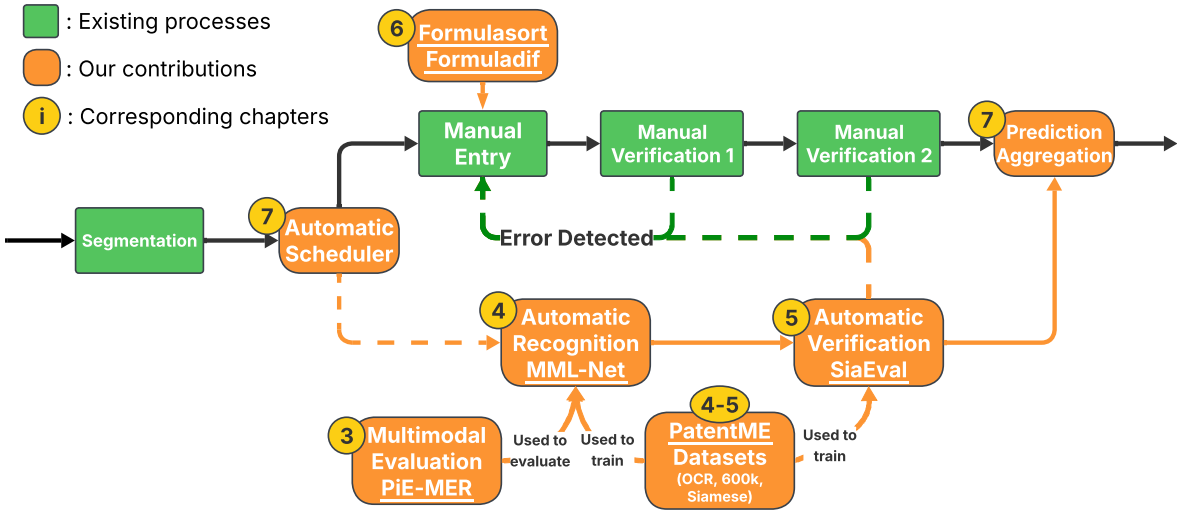
More specifically, thanks to the collaboration with the company Luminess in this PhD, we narrow our scope to printed mathematical expressions extracted from patent applications submitted to the European Patent Office (EPO).

When an inventor submits a patent application, the input document can be noisy, of variable quality, and may follow its own formatting and document standards. To perform prior-art search, information retrieval, and long-term archiving, Luminess converts, for the EPO, the patent applications from image format into normalized XML and PDF documents that provide a textual and highly standardized representation of the original documents. The required conversion accuracy is 99.99%, as even a single error can significantly alter the meaning of a patent and potentially have serious legal consequences. Luminess currently relies on a mostly manual pipeline to convert the documents while maintaining this extremely high level of quality (see Section 1.2). Mathematical expressions are so complex that they are processed separately through their own dedicated processing pipeline.

The objective of this thesis is to study current mathematical OCR systems, measure their performance for a potential integration into the Luminess production pipeline, and investigate how the recognition process can be accelerated while maintaining or improving the quality achieved by the current manual workflow. We also aim to reduce the tediousness of manually transcribing mathematical expressions for human annotators. The PhD thesis is structured into eight chapters: this introduction, a state-of-the-art chapter, four chapters presenting the main contributions, a chapter that combines all the contributions and proposes a new mathematical expression recognition pipeline, and a general conclusion. A detailed description of each chapter is provided below. Figure 1.2 summarizes the main contributions of this thesis and the chapters in which they are presented, using a diagram that shows the proposed processing pipeline in comparison with the current manual recognition pipeline.



(a) The existing Luminess pipeline for the manual transcription of mathematical expressions.



(b) Proposed pipeline combining the contributions developed throughout the thesis, with the corresponding chapter numbers indicated on the different components.

Figure 1.2: Overview of the existing and proposed mathematical expression recognition pipelines.

- **Chapter 1 – General Introduction.** This chapter presents the general context of mathematical expression recognition, the scientific and industrial challenges underlying the research problem addressed in this thesis, and the specific industrial context of the work. It also presents the mathematical expression transcription pipeline currently used at Luminess, the industrial partner of this CIFRE PhD.

- **Chapter 2 – State of the Art: Models, Datasets, and Metrics for Mathematical Expression Recognition** This chapter reviews the evolution of Mathematical Expression Recognition, covering the evolution of the main approaches, from early computer vision-based approaches to recent Vision-Language Models, as well as their training datasets, output representations, and evaluation metrics. The current state of the art is reviewed with a particular focus on the limitations of existing systems in the context of real-world mathematical expression recognition in patents.

- **Chapter 3 – PiE-MER: a Robust Multimodal Mathematical Expression Recognition Evaluation Pipeline** This chapter presents PiE-MER, a multimodal evaluation framework developed after identifying limitations in existing approaches to evaluating mathematical OCR systems. It provides a broad and more rigorous evaluation of OCR predictions by combining multiple evaluation modalities and metrics. We use it to evaluate state-of-the-art OCR systems on multiple datasets and show that they generally struggle outside of their training corpus, which raises concerns about their use on patent data.

- **Chapter 4 – PatentME-600k and MML-Net: A Large-Scale Dataset and Model for Patent Mathematical Expression Recognition** Motivated by the limitations identified in Chapter 3, this chapter investigates whether adapting both the training data and the output representation to the patent domain can improve mathematical expression recognition. Existing systems mainly generate LaTeX and are trained on datasets that do not necessarily reflect the heterogeneous visual characteristics of patent documents, including variations in fonts, image quality, resolution, and noise. The PatentME-600k dataset is introduced and used to train MML-Net, a dedicated Vision Transformer Encoder-Decoder model on 600,000 patent image–MathML pairs. Although the resulting model is better adapted to the targeted domain and achieves good performance, its accuracy remains insufficient to meet the 99.99% quality required for fully autonomous recognition.

- **Chapter 5 – SiaEval: a Reference-Free Post-OCR Verification Task for Printed Mathematical Expression Recognition** This chapter introduces SiaEval, a reference-free post-OCR verification approach based on a Siamese architecture. The objective is to determine whether OCR predictions are correct by comparing the original patent image with a rendering of the predicted MathML, without requiring the ground-truth transcription at inference time. The OCR errors produced by the model developed in the previous chapter are used to construct realistic training pairs. Two datasets are introduced: PatentME-OCR, used to generate realistic OCR predictions and errors, and PatentME-Siamese, used to train and evaluate the verification model. A conservative confidence threshold allows highly reliable predictions to be accepted automatically, while uncertain cases are redirected to human verification. The results demonstrate that OCR errors can be detected with high precision in a production setting using SiaEval.
- **Chapter 6 – Formulasort and Formuladif: Facilitating Manual Transcription of Mathematical Expressions** Since fully automatic recognition cannot be expected to achieve the required quality for all mathematical expressions, human transcription remains necessary for a subset of cases. This chapter aims at making the remaining human work more efficient and more reliable. It presents Formulasort and Formuladif, two tools designed to simplify the manual transcription process and reduce transcription errors. Formulasort organizes mathematical expressions according to their visual similarity to make the transcription quicker and more accurate, while Formuladif generates difference images between successive expressions to highlight subtle changes. The results show that they can be effective at better organizing the transcription workload to reduce errors.
- **Chapter 7 – Integration of Our Contributions into a New Semi-Automatic Recognition Pipeline** This chapter brings together the contributions developed throughout the thesis into a semi-automatic recognition pipeline designed to assist the existing manual transcription workflow at Luminess. We propose a pipeline combining the patent-specific MML-Net model, SiaEval, and Formulasort. An automatic scheduler is trained to determine whether each incoming mathematical expression should be processed by the automatic OCR pipeline or by the human pipeline. Our objective is to increase the overall quality of the process, identify potentially erroneous OCR predictions, and reduce the manual workload while maintaining the extremely high accuracy required for mathematical expression recognition in patent applications.
- **Chapter 8 – Discussion and Conclusions.** This final chapter discusses the main findings and contributions of the thesis, their limitations, and the perspectives opened by the proposed approaches. It concludes by discussing the remaining challenges for the reliable automation of mathematical expression recognition in patent documents and the potential future developments of the proposed pipeline.

1.2 Industrial Context and Motivation

1.2.1 A Short History of Intellectual Property

A patent is a technical document that gives its authors a monopoly over using and selling their invention for a limited period of time, in exchange for disclosing the design so that a sufficiently skilled person could replicate it. It consists of a detailed description of the invention, and can contain diagrams, mathematical expressions, and tables that explain its principles and usage. There is also a list of claims, which are very precise legal statements that define what exactly is protected by the patent and cannot be reproduced.

Patents are published and made available by patent offices on their online portal. This allows the public to access all published patents and build upon existing knowledge to create new innovations. Patent databases are essential to drive progress in most industrial sectors. According to the World Intellectual Property Organization, more than 3.7 million patent applications were filed worldwide in 2024 across all patent offices. Depending on the office, between about 50% and nearly 100% of processed applications resulted in a patent grant. [3].

One of the first known notion of intellectual property granting an exclusive commercial advantage can be traced back to Antiquity. In the *Deipnosophists* [4], Athenaeus describes a rule according to which cooks who invented a new and successful dish were granted exclusive rights to produce it and benefit from its profits for one year:

> *And if any caterer or cook invented any peculiar and excellent dish, no other artist was allowed to make this for a year; but he alone who invented it was entitled to all the profit to be derived from the manufacture of it for that time; in order that others might be induced to labour at excelling in such pursuits.*

The purpose of this early form of intellectual property was already to encourage innovation by granting inventors a temporary commercial advantage in return for their creative efforts.

The idea later reappeared in medieval Europe in a more formalized form. In 1421, the Florentine architect Filippo Brunelleschi was granted an exclusive privilege for his invention of a vessel designed to transport marble on the Arno River, generally considered one of the earliest documented patent-like privileges[4].

In the following centuries, such privileges were commonly granted through *letters patent*, which were royal grants giving particular rights or privileges to their recipients.

[3] https://www.wipo.int/edocs/pubdocs/en/wipo-pub-941-2025-en-world-intellectual-property-indicators-2025.pdf, accessed 2026-08-17.

[4] https://www.wilsongunn.com/history/history_patents.html, accessed 2026-08-17.

They could, for example, grant individuals exclusive rights to produce or sell a particular product or to practice a particular invention within a territory.

These privileges were eventually regulated to limit the granting of monopolies. In England, the *Statute of Monopolies* of 1624 restricted royal monopolies while preserving a limited exception for new inventions[5].

National patent systems were then progressively established. A major step was the creation of the European Patent Convention (EPC) in 1973, which established a common framework for granting European patents[6]. The convention led to the creation of the EPO, which is responsible for examining and granting all European patents.

1.2.2 The Patent Publication Process at the European Patent Office

For a patent to be granted, it must follow a very strict list of requirements that depends on the patent office where it is filed. The three key rules for patentability are that the invention needs to be new (never disclosed to the public before the submission date), be capable of industrial application, and involve an inventive step (an expert could not have easily inferred the same idea). More than 4,000 patent examiners[7] review them in detail by checking previous patents and scientific literature to ensure novelty, and give their expert opinion on the inventiveness of the patent.

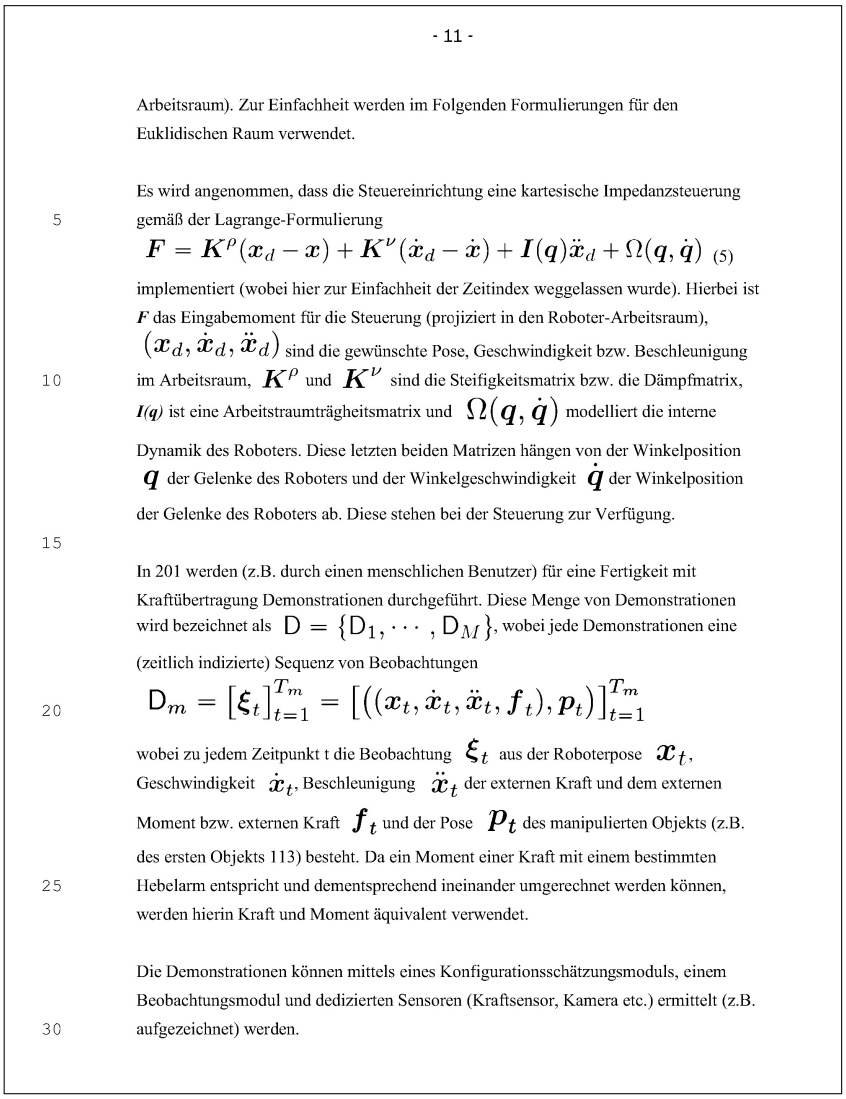
Patent applications are published before the examination procedure is completed, regardless of whether the patent is ultimately granted. If granted, they are made available as final standardized patent documents, following strict specifications concerning page layout, image sizes and formats, tables, and mathematical and chemical expressions. For the EPO specifically, the claims section also needs to be translated into the three official languages (English, French, and German) [8].

This page formatting process is not trivial. Patent applications are written by multiple different actors, such as the inventors themselves, the intellectual property teams of large companies, or private patent attorneys and engineers hired by smaller companies to help them write the document. Patent offices provide some guidelines for drafting, but authors are still fairly free in terms of format and style as long as they respect the conventions in place [9]. As a result, there is a strong need to convert these heterogeneous patent applications into a standardized format. This task is particularly challenging due to the large variability in document layouts, writing styles, and overall quality across patent applications. This heterogeneity creates an important domain

[5] https://en.wikipedia.org/wiki/Statute_of_Monopolies, accessed 2026-08-17.
[6], accessed 2026-08-17.
[7] https://www.epo.org/en/news-events/press-centre/fact-sheet/446602, accessed 2026-09-11.
[8] https://www.epo.org/en/legal/epc/2020/a14.html, accessed 2026-08-17.
[9] https://www.epo.org/fr/legal/official-journal/2026/03/a19, accessed 2026-08-17.

gap between the datasets commonly used to develop and evaluate recognition systems and the documents encountered in real-world patent processing. An example patent application page is shown in Figure 1.3.

To fully standardize these documents and produce high-quality, searchable patent documents, especially when building a critical database containing all European patent publications, the normalization process starts from the original image-based document rather than relying on the heterogeneous formats and structures provided by the applicants. This makes it possible to apply the same processing and formatting rules to every patent, regardless of how the original application was produced, and to build a consistent and searchable database of European patents.



- 11 -

Arbeitsraum). Zur Einfachheit werden im Folgenden Formulierungen für den Euklidischen Raum verwendet.

Es wird angenommen, dass die Steuereinrichtung eine kartesische Impedanzsteuerung gemäß der Lagrange-Formulierung

$$\boldsymbol{F} = \boldsymbol{K}^{\rho}(\boldsymbol{x}_d - \boldsymbol{x}) + \boldsymbol{K}^{\nu}(\dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}}) + \boldsymbol{I}(\boldsymbol{q})\ddot{\boldsymbol{x}}_d + \Omega(\boldsymbol{q}, \dot{\boldsymbol{q}}) \quad (5)$$

implementiert (wobei hier zur Einfachheit der Zeitindex weggelassen wurde). Hierbei ist $\boldsymbol{F}$ das Eingabemoment für die Steuerung (projiziert in den Roboter-Arbeitsraum), $(\boldsymbol{x}_d, \dot{\boldsymbol{x}}_d, \ddot{\boldsymbol{x}}_d)$ sind die gewünschte Pose, Geschwindigkeit bzw. Beschleunigung im Arbeitsraum, $\boldsymbol{K}^{\rho}$ und $\boldsymbol{K}^{\nu}$ sind die Steifigkeitsmatrix bzw. die Dämpfmatrix, $\boldsymbol{I(q)}$ ist eine Arbeitstraumträgheitsmatrix und $\Omega(\boldsymbol{q}, \dot{\boldsymbol{q}})$ modelliert die interne Dynamik des Roboters. Diese letzten beiden Matrizen hängen von der Winkelposition $\boldsymbol{q}$ der Gelenke des Roboters und der Winkelgeschwindigkeit $\dot{\boldsymbol{q}}$ der Winkelposition der Gelenke des Roboters ab. Diese stehen bei der Steuerung zur Verfügung.

In 201 werden (z.B. durch einen menschlichen Benutzer) für eine Fertigkeit mit Kraftübertragung Demonstrationen durchgeführt. Diese Menge von Demonstrationen wird bezeichnet als $\mathsf{D} = \{\mathsf{D}_1, \cdots, \mathsf{D}_M\}$, wobei jede Demonstrationen eine (zeitlich indizierte) Sequenz von Beobachtungen

$$\mathsf{D}_m = \left[\boldsymbol{\xi}_t\right]_{t=1}^{T_m} = \left[\left((\boldsymbol{x}_t, \dot{\boldsymbol{x}}_t, \ddot{\boldsymbol{x}}_t, \boldsymbol{f}_t), \boldsymbol{p}_t\right)\right]_{t=1}^{T_m}$$

wobei zu jedem Zeitpunkt t die Beobachtung $\boldsymbol{\xi}_t$ aus der Roboterpose $\boldsymbol{x}_t$, Geschwindigkeit $\dot{\boldsymbol{x}}_t$, Beschleunigung $\ddot{\boldsymbol{x}}_t$ der externen Kraft und dem externen Moment bzw. externen Kraft $\boldsymbol{f}_t$ und der Pose $\boldsymbol{p}_t$ des manipulierten Objekts (z.B. des ersten Objekts 113) besteht. Da ein Moment einer Kraft mit einem bestimmten Hebelarm entspricht und dementsprechend ineinander umgerechnet werden können, werden hierin Kraft und Moment äquivalent verwendet.

Die Demonstrationen können mittels eines Konfigurationsschätzungsmoduls, einem Beobachtungsmodul und dedizierten Sensoren (Kraftsensor, Kamera etc.) ermittelt (z.B. aufgezeichnet) werden.

Figure 1.3: A page from an actual patent application submitted to the EPO.

1.2.3 Luminess Patent Conversion Pipeline

It is the role of Luminess, the company partner of this CIFRE thesis, to perform the conversion and normalization of all these patent applications from image format into normalized documents, both to allow patent experts to perform their work and to populate the EPO patent database, as illustrated in Figure 1.4. Luminess has been performing this work with the EPO since 1990 and has developed a customized process for converting patent applications into their final document format. The conversion accuracy required by the contract with the EPO is 99.99%, meaning that the process must be highly reliable to avoid introducing any errors into the description or claims of the final document of the invention, which could have very serious legal consequences. The processing time must also be very short, as a large number of patent applications need to be processed: 201,974 applications were filed in 2025, corresponding to approximately 550 applications per day [10].

Patent documents are also often very long and dense, meaning that the complete conversion process must be performed quickly while maintaining a very high level of accuracy. These requirements require finding a balance between automation, which improves scalability but may introduce errors, and manual transcription, which ensures quality but remains costly and difficult to scale.

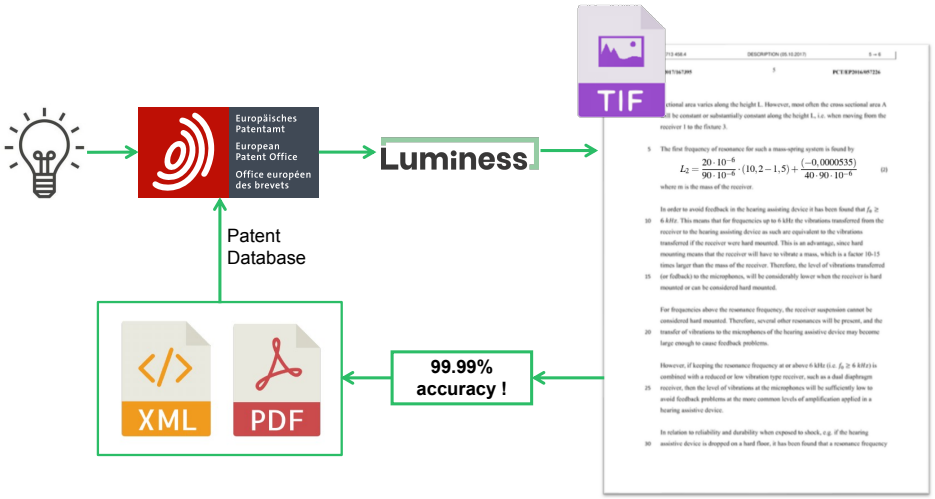


Figure 1.4: The conversion of patent applications to normalized XML and PDF documents.

The existing conversion pipeline begins with a manual segmentation of the input pages to separate the document into several categories, in reading order: text paragraphs, tables, chemical formulas, mathematical formulas (Figure 1.5), figures, titles and more. This first step serves as an initial verification of the input document and its overall

[10] https://www.epo.org/en/news-events/press-centre/fact-sheet/446602, accessed 2026-09-11.

quality, detecting potentially unusable documents, and as a preprocessing stage to rotate pages that are upside down or incorrectly oriented, making the subsequent OCR tasks more reliable.

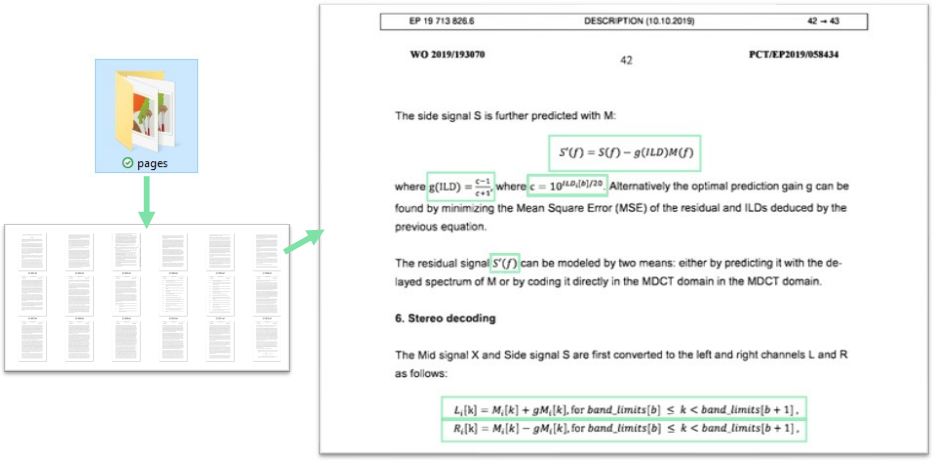


Figure 1.5: An illustration of the segmentation of mathematical expression images from patent applications.

Once the document has been segmented in reading order, the pipeline applies a dedicated processing step to each category. Text regions are sent to the company's proprietary multi-OCR system, figures are annotated with their figure and part numbers, while mathematical expressions are forwarded to a dedicated manual transcription process in which human operators manually transcribe each formula.

The mathematical expressions are manually transcribed in MathML format using a MathType[11] virtual keyboard. A dedicated interface displays the mathematical expression together with part of its surrounding context, allowing the operator to enter the formula while checking that the entire mathematical content has been selected.

The transcribed mathematical expressions then go through two successive manual verification steps to achieve the required quality of 99.99%. For each verification step, a human operator compares the original image with the rendered version of the transcription and must confirm that no mistake was made. If an error is detected, the expression is returned to the operator who originally transcribed it to correct the error and resubmit the expression for another verification. This recognition pipeline is illustrated in Figure 1.2a. This process is designed to avoid the most serious case: an inventor detecting an error in the final document and requesting what is referred to as a "reprocess", meaning that the entire patent must be processed again. Avoiding such full reprocessing is a major objective of the pipeline. The final output of the patent conversion pipeline consists of an XML document, a PDF, and the associated images of mathematical expressions, chemical expressions and figures, all stored in a zip archive on the EPO patent portal.

[11] https://www.wiris.com/en/mathtype/, accessed 2026-08-17.

1.3 Research Questions

The overall objective of this thesis is to study how the recognition of mathematical expressions in heterogeneous technical documents can be progressively automated while maintaining the high level of quality required in real-world applications. In particular, the thesis addresses the following research questions:

- **RQ1:** *What are the capabilities and limitations of current MER methods when applied to real-world patent documents?*
- **RQ2:** *How can the quality and reliability of MER predictions be measured, both during model evaluation and in production?*
- **RQ3:** *How can large-scale technical document databases be used to improve automatic MER?*
- **RQ4:** *How can automatic systems assist human operators by reducing processing time, errors, and cognitive load?*
- **RQ5:** *Under which conditions can MER be reliably automated, and when is human intervention still required?*

Answering these questions requires studying the complete digitization process, from automatic recognition and quality evaluation to error detection and human assistance. Our goal is to improve recognition accuracy, measure the reliability of automatic predictions, and determine when human intervention is still required to ensure the required conversion quality.

Chapter 2

State of the Art: Models, Datasets, and Metrics for Mathematical Expression Recognition

The goal of this state-of-the-art chapter is to give a historical overview of the research conducted on Mathematical Expression Recognition (MER) and to highlight the main challenges identified in the literature that motivate the contributions presented in this thesis. This chapter provides a reference for the architectures, datasets, and evaluation protocols used throughout the manuscript.

The three following sections focus on MER models, datasets, and evaluation metrics, respectively. Within each section, the presented works are mainly organized chronologically to highlight the evolution of the field over time.

2.1 Mathematical Expression Recognition Models

This first section summarizes the architectures used for offline MER. We begin with the earliest models, which separated symbol and layout recognition in two distinct steps. We then present the first encoder-decoder models, followed by the emergence of Transformer and Vision Transformer end-to-end architectures, and finally recent vision-language models.

2.1.1 Early Computer Vision Approaches

The main differences between plain text recognition and MER lie in the large variety of mathematical symbols that compose the vocabulary and in the complex two-dimensional spatial organization of mathematical expressions. In particular, variations in character size and position can alter the meaning of an expression. That is why the earliest approaches to MER separated the problem into two consecutive tasks: the recognition of individual symbols and the understanding of the layout.

For readers looking for a quick overview of these approaches, these two surveys describe the main ideas behind symbol recognition and structural analysis, and review much of the foundational work [5, 6].

The earliest work we identified using a two-step strategy is [7]. Anderson used a RAND tablet (Figure 2.1) to collect handwritten symbols and combined character recognition from [8] with rule-based layout analysis to reconstruct mathematical expressions. This two-step strategy was later extended to different types of mathematical documents, including printed documents and PDF, using increasingly sophisticated rule-based layout analysis [9, 10, 11, 12].

The INFTY OCR system [13, 14] extended this approach to full-page mathematical documents by combining document segmentation, symbol recognition, rule-based layout analysis, and manual correction. It achieved an Exact Match rate of 89.6% on 12,493 expressions.

Later work improved the symbol-recognition component using machine-learning

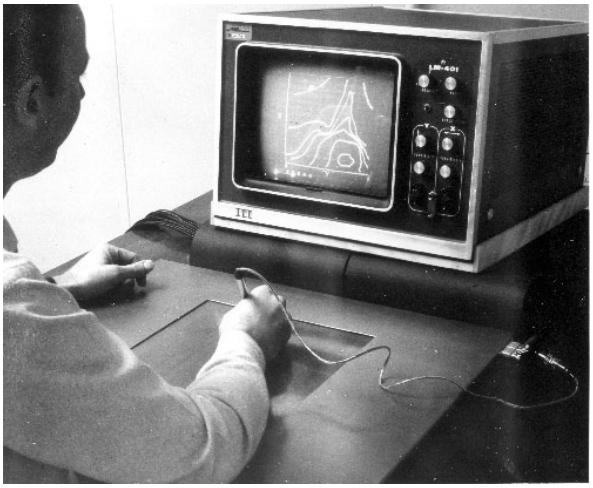


Figure 2.1: The RAND tablet used in [7] for handwritten MER.

error classifiers, including Support Vector Machines, while still using the separation between symbol recognition and structural analysis [15, 16, 17, 18, 19]. In 2006, [20] introduced a dataset of 43,495 annotated mathematical expressions and a multi-stage system that combined symbol segmentation and recognition with rule-based layout parsing. However, the dataset was not publicly available, limiting reproducibility and cross-system comparison.

Overall, these multi-stage approaches were widely used between the 1990s and the early 2010s, and demonstrated promising performance at the time. However, they were generally evaluated on independent datasets using different evaluation protocols and metrics, making direct comparisons difficult. Moreover, they were often highly optimized for the specific use case and data considered in each work, making their ability to generalize uncertain. In addition, most of the original implementations are no longer publicly available, making reproducibility difficult.

Those limitations motivated end-to-end machine-learning-based approaches, which aimed to jointly learn symbol recognition and the relationships between symbols to produce structured textual mathematical representations.

2.1.2 Encoder–Decoder Architectures

The limitations of the two-step pipelines described above, particularly error propagation and handcrafted layout parsers, motivated the development of end-to-end trainable architectures. Instead of separately recognizing symbols and analyzing their structure, these approaches encode the input image into visual features and directly decode them into a sequence of tokens, typically LaTeX.

While encoder–decoder models had already been explored extensively for online handwritten MER [21], [22] marked an important step for offline printed MER (PMER). Deng et al. introduced the im2latex100k dataset, one of the first large-scale datasets

pairing mathematical expression images with their LaTeX representations. This dataset enabled the training of larger neural models and provided a common benchmark for subsequent work.

Building on image captioning architectures [23, 24], they proposed an encoder–decoder model for image-to-LaTeX conversion. A CNN extracts visual features, which are enriched with positional information using an LSTM, while a second LSTM decodes the resulting representation into LaTeX tokens. They also introduced a coarse-to-fine attention mechanism that first identifies relevant image regions before computing fine-grained attention within them.

For evaluation, the authors combined BLEU [25] with a visual exact-match metric based on rendered expressions, addressing the fact that different LaTeX sequences can produce identical mathematical renderings (see Section 2.3). Their best model achieved approximately 88% BLEU and 77% Visual Exact Match.

The importance of this work lies in its architecture, proposed dataset and evaluation protocol. The im2latex100k dataset became a standard benchmark for PMER and supported a large quantity of encoder–decoder research. This contrasts with earlier approaches, which generally relied on smaller, independently developed datasets and evaluation protocols.

This work triggered a large number of publications on encoder–decoder MER models, with studies mainly improving the CNN encoder, attention mechanisms, training strategies, or decoding process. Despite these variations, most retained the general architecture illustrated in Figure 2.2. The resulting research can be grouped into several directions, including improvements to visual feature extraction, spatial representation and attention, as well as sequence level training and decoding. The main contributions and reported performance of the approaches discussed in the following section are summarized in Table 2.1.

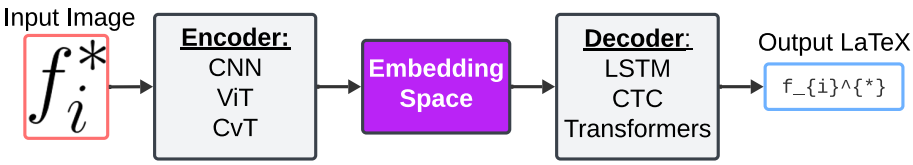


Figure 2.2: Simplified encoder–decoder architecture used for image-to-LaTeX conversion. The encoder extracts visual features from the input image, while the decoder generates the corresponding LaTeX sequence token by token from the embedded formula.

Incremental Improvements to Deng et al. Encoder–Decoder Architecture

Several works introduced incremental modifications to the architecture of [22]. [26] used beam search during inference, removed the LSTM row encoder, and modified the CNN encoder, slightly improving recognition performance. [27] also removed the encoder-side

Paper	Encoder Improvement	Decoder Improvement	Other Improvements	BLEU (%)	Exact Match (%)	Edit Score (%)
[22]	CNN + row LSTM for spatial awareness	LSTM + hierarchical coarse-to-fine attention	Introduced the im2latex100k dataset	87.73	77.46^{v}	–
Incremental Improvements to the Encoder–Decoder Architecture						
[26]	CNN without max-pooling	–	Beam search at inference	78.00	35.00	76.00^{t} 62.00^{v}
[27]	CNN with two pooling variants	Deep Output Layer instead of LSTM	–	89.00	70.37^{t}	93.24^{t}
[28]	Enhanced CNN with more layers and less dimensionality reduction, Max-Blur-Pooling for shift invariance	–	–	89.05	87.74^{v}	92.35^{v}
Improving Spatial Representation and Attention						
[29]	ResNet-inspired CNN	–	Spotlight mechanism for region tracking	–	–	77.80^{t}
[30]	DenseNet larger encoder with channel-wise and spatial attention	–	–	88.25	37.09^{t}	91.57^{t}
[31]	CNN-LSTM with additional conv/pooling layers and double-attention	–	–	88.42	79.81^{v}	88.57^{t}
Sequence-Level Training and Alternative Decoding Strategies						
[32]	CNN + positional encoding	–	Sequence-level RL with self-fed tokens	90.28	82.33^{v}	92.28^{v}
[33]	CNN + positional encoding	–	Random padding on the images + RL	94.07	79.1^{v} 43.6^{t}	–
[34]	ResNet encoder	Convolutional decoder	Fully convolutional architecture	88.33	83.41^{v}	90.80^{v}
[35]	CNN + RNN encoder	Encoder-only	CTC decoding	89.4	84.79^{t}	96.54^{t}

Table 2.1: Summary of results on the im2latex100k dataset for encoder–decoder architectures. The reported BLEU score, the stricter Exact Match when available, and the edit score are shown. Edit scores are reported either at the visual level (^{v}) or at the textual level (^{t}).

LSTM and replaced the decoder with a Deep Output Layer [36] that combines the LSTM state, attention vector, and previous token embedding. [28] focused on the CNN, adding extra layers, limiting downsampling, and using Max-Blur-Pooling [37] to improve robustness to small spatial shifts.

Improving Spatial Representation and Attention

Other works focused on providing the decoder with better spatial information. [29] introduced a spotlight mechanism that tracks previously attended regions and predicted tokens, helping the model learn a reading order for two-dimensional expressions. [30] used a larger DenseNet encoder with channel-wise and spatial attention, while [31] introduced an additional convolutional layer and a double-attention mechanism. These approaches highlight the importance of explicitly modeling spatial relationships, since mathematical expressions cannot always be read from left to right.

Sequence-Level Training and Alternative Decoding Strategies

Another line of work addressed the training objective and decoding strategy. [32] combined 2D sinusoidal positional encoding with a two-stage training procedure: token-level training followed by sequence-level reinforcement learning using a BLEU-based reward. The model also did not use Teacher Forcing during the training stage to reduce exposure bias. [33] followed a similar direction, combining positional encoding, random padding, and token-level reinforcement learning. These approaches reported the highest BLEU scores among the models discussed in this section (Table 2.1).

Alternative decoding strategies were also explored. ConvMath [34] replaced the recurrent LSTM decoder with a fully convolutional decoder [38] and used a ResNet encoder. Py4MER [35] instead combined a CNN–BiLSTM encoder with Connectionist Temporal Classification (CTC), learning the alignment between visual features and the target LaTeX sequence without an autoregressive decoder. These works demonstrate that LSTM-based decoding was not the only viable approach to image-to-LaTeX generation.

These studies illustrate the limitations of encoder–decoder architectures. Effective recognition relies heavily on visual feature extraction, spatial information, attention mechanisms, and sequence generation.

The next major architectural upgrade came with Transformer-based models, which provided a different way to model the long-range dependencies and spatial relationships present in mathematical expressions.

2.1.3 Transformer-based Models

Unlike the CNN–RNN encoder–decoder models discussed above, recent approaches rely on Transformer-based architectures for visual encoding, decoding, or both. Transformers [39] use self-attention to model relationships between their input tokens. The Vision Transformer (ViT) [40] applies this principle to images by representing them as sequences of patches.

Many variants have been developed for more complex visual data. The Swin Transformer [41] limits self-attention to local windows and shifts them between layers to reduce the computational cost while allowing information exchange across windows. The Convolutional Vision Transformer (CvT) [42] combines convolutional layers with Transformer layers to use both the local feature extraction of CNNs and the attention of Transformers.

Transformers have also become widely used for document image understanding. Models such as Pix2Struct [43] and Donut [44] perform document understanding directly from images, avoiding the traditional OCR-first pipeline and its associated error propagation. Related architectures have been applied to text recognition with TrOCR [45] and to academic document understanding with Nougat [46].

Self-attention is particularly suited to mathematical expressions because the interpretation of a symbol depends on its spatial and structural relationships with potentially distant symbols. This allows Transformer-based models to capture dependencies between superscripts, subscripts, fractions, and other structural operators more directly than architectures relying on local visual features. However, Transformers generally require a large amount of training data to learn effective visual representations. That is why Transformer-based approaches rely on large-scale datasets, pretraining, or data augmentation to compensate for limited training data. The Transformer-based encoder–decoder models discussed in this section are summarized in Table 2.2.

Early Transformer-Based Encoder–Decoder Architectures

An early Transformer-based approach to PMER is pix2tex [47]. It combines a ResNetV2 encoder, a Vision Transformer, and a Transformer decoder. The model is trained on a mix of rendered and handwritten expressions from im2latex100k, CROHME [60], and additional online data, with multiple fonts used for data augmentation. Its mixed training and evaluation data make direct comparison with models evaluated exclusively on im2latex100k difficult.

Another contribution is a hybrid CNN–Transformer architecture in [48], which combines a ResNet encoder, 2D positional encoding, a Vision Transformer, and an LSTM decoder with coverage attention. These early approaches marked the transition from CNN–RNN architectures towards Transformer-based visual encoding and decoding.

Paper	Encoder Improvement	Decoder Improvement	Other Improvements	BLEU (%)	Exact Match (%)	Edit Score (%)
Early Transformer-Based Encoder–Decoder Architectures						
[47]	ResNetV2 + Vision Transformer	Transformer	Mixed rendered and handwritten training data	88.00	60.00^{t}	90.00^{t}
[48]	ResNet + Vision Transformer	LSTM with coverage attention	2D positional encoding	48.39	89.94^{t} 92.23^{v}	86.48^{v}
Bidirectional and Alternative Transformer Decoding						
[49]	DenseNet + 2D positional encoding	Bidirectional Trained Transformer	Bidirectional prediction (L-to-R and R-to-L)	–	67.91^{t}	95.08^{t}
[50]	CNN + Transformer encoder	CTC module	Encoder-only, non-autoregressive decoding	90.50	49.00^{t} 84.40^{v}	–
Improving Visual Representations with Local and Global Information						
[51]	ConvMixer + Conditional Network	Transformer with conditional token injection	Global image information injected into the decoder	93.53	93.20^{v}	–
[52]	Symbol-level CNN + positional encoding + self-attention	Transformer	Connected-component segmentation and symbol-level features	92.93	89.00^{v}	–
[53]	Dual-branch encoder (local CNN + global branch)	Transformer	Context Coupling Module + Dynamic Soft Target	92.90	89.71^{v} 90.01^{v} (w/o space)	–
[54]	Dual-branch Focal Transformer + CNN	Transformer	Fusion Enhancement Module and multi-scale feature fusion	97.00	89.90^{v} 90.05^{v} (w/o space)	–
[55]	Hierarchical Vision Transformer	Transformer	Sub-formula decomposition and hierarchical supervision	–	98.20^{v} (I2L) 96.30^{v} (UniMER)	96.80^{v} (HDR-Test)

Table 2.2: Summary of transformer-based MER models. Results are reported on the datasets specified by the corresponding papers. BLEU scores, Exact Match rates, and edit-based metrics are reported when available. Textual and visual metrics are denoted by ^{t} and ^{v}, respectively.

Paper	Encoder Improvement	Decoder Improvement	Other Improvements	BLEU (%)	Exact Match (%)	Edit Score (%)
Scaling Data and Training						
[56]	Customized Swin Vision Transformer	mBART	Variable-input preprocessing and aspect-ratio preservation	84.23	–	93.49^{t}
[57]	Vision Transformer (300M parameters)	Transformer	Large-scale Tex80M dataset with extensive data augmentation	$93.80^{\dagger}$	–	–
[58]	Convolutional Vision Transformer	Transformer	im2latexv2: 59 fonts and 600 DPI rendering	94.50	–	94.70^{v}
[59]	Swin Transformer	mBART	Length-Aware Module and diverse data augmentation	$91.60^{\ddagger}$	–	94.00^{t}

Table 2.2: (Continued): Summary of transformer-based MER models (continued). Results are reported on the evaluation datasets specified by the corresponding papers. † TexTeller: EPMR metric on Tex80M-Test. ‡ UniMERNet: results on complex printed expressions from UniMER-Test, with the edit score reported using textual edit distance. Textual and visual metrics are denoted by ^{t} and ^{v}, respectively.

Bidirectional and Alternative Transformer Decoding

Several works explored alternatives to conventional left-to-right autoregressive decoding. [49] adapted the Bidirectional Trained Transformer (BTTR) from handwritten MER to printed expressions. Using a DenseNet encoder with 2D positional encoding, the model learns expressions in both left-to-right and right-to-left directions, allowing each prediction to benefit from information from both sides of the expression. The authors also increased the frequency of rare symbols during training using the Marmot dataset and tools [61, 62].

CTC, previously used in Py4MER [35], was combined with a CNN–Transformer encoder in [50]. The resulting encoder-only architecture extracts local and global visual features and uses CTC for non-autoregressive transcription. These works demonstrate that Transformer-based MER can move beyond conventional autoregressive decoding through bidirectional generation and CTC-based transcription.

Improving Visual Representations with Local and Global Information

Other approaches focused on combining fine-grained local information with the global structure of the expression. ICCT [51] uses a ConvMixer [63] encoder and injects global image information in the Transformer decoder through a conditional token. EDSL [52] takes the opposite approach by explicitly extracting individual symbols using a simple connected-component analysis. Each symbol is encoded with a CNN, positional encoding, and self-attention before being processed by a Transformer decoder. Building on EDSL, DBN [53] combines local symbol features with a global representation of the complete expression through cross-attention and additional supervision.

TRMER [54] removes the need for explicit symbol segmentation while using the local–global principle. Its Dual-Branch Encoder combines a Focal Transformer [64] with a convolutional branch for multi-scale spatial features. Finally, HDNet [55] introduces hierarchical supervision by processing both complete formulas and sub-formulas with a Vision Transformer encoder. The authors also introduce HDR-100M, containing more than 100 million training samples, and the multi-annotated HDR-Test dataset.

Overall, these approaches progressively explored different ways of combining local visual details with global structural information while preserving the structure of the expression.

Scaling Data and Training

Recent work has increasingly shifted from architectural modifications towards scaling and diversifying the training data.

Texify [56] uses a Swin ViT encoder and an mBART decoder [65], inspired by Donut [44]. Unlike most dedicated PMER models, it is designed to recognize images containing

both text and mathematical expressions. It is trained on im2latex100k and additional data from arXiv papers, and was later integrated into the Surya document recognition system [66].

TexTeller [57] uses a relatively simple 300M-parameter ViT–Transformer architecture and focuses primarily on large-scale training. Its Tex80M dataset is built from more than 300,000 scientific papers and combines multiple forms of augmentation, including formula concatenation, symbol replacement, inserted text, font variation, and realistic paper textures. The model also includes a small proportion of handwritten expressions. This work demonstrates that a relatively simple architecture can achieve strong performance when trained on sufficiently large and diverse data.

MathNet [58] investigates the importance of rendering diversity. Its im2latexv2 dataset extends im2latex100k by rendering expressions with 59 fonts at 600 dots per inch (DPI). The authors show that models trained on a single visual distribution generalize poorly to other rendering conditions, while multi-font training improves robustness. The results also highlight the importance of input resolution on the performance.

UniMERNet [59] extends this data-centric approach to a broader range of real-world conditions. Its UniMER dataset includes standard and complex printed expressions, screen-captured formulas, and handwritten expressions. Trained on the large-scale UniMER-1M dataset, it is evaluated across the different UniMER-Test subsets, providing a broader evaluation of robustness than traditional printed-expression benchmarks.

These works illustrate the evolution from model-centric to data-centric improvements. The scale, diversity, resolution, and realism of the training data have become as important as architectural complexity for improving MER robustness.

2.1.4 Vision-Language Models for MER

The recent development of large-scale Vision-Language Models (VLMs) has impacted document analysis and recognition. Unlike specialized MER models, VLMs are designed to process a broad range of visual and textual tasks, including OCR, document understanding, visual question answering, and mathematical reasoning.

Several general-purpose proprietary VLMs have demonstrated strong performance across a wide range of multimodal tasks. These include GPT-4V and GPT-4o [67], Gemini[1], and Claude[2]. Open-weight models such as Qwen3-VL [68] have also demonstrated strong performance on general multimodal and visual mathematical benchmarks. However, these models are not specifically designed for MER, and their training data and training procedures are generally not fully disclosed.

[1] https://blog.google/products-and-platforms/products/gemini/gemini-3/, accessed 2026-07-30.

[2] https://www.anthropic.com/news/claude-3-family, accessed 2026-07-30.

Issues with Using General-Purpose VLMs for MER

Strong performance on general-purpose or visual mathematical reasoning benchmarks does not necessarily imply accurate mathematical expression transcription. For example, MathVista [69] and MathVerse [70] evaluate mathematical reasoning and visual understanding rather than exact transcription into representations such as LaTeX. A model may be able to solve a mathematical problem correctly while producing an inaccurate transcription of the expression.

The task considered in this thesis is narrower: mathematical expressions are already segmented, and the objective is to transcribe each expression accurately. Layout analysis and recognition of the rest of the document is unnecessary. This makes direct comparison with general-purpose VLMs difficult because many are evaluated on heterogeneous document or reasoning benchmarks and not on MER directly. Formula-specific evaluation is required to compare them with dedicated PMER models.

Some studies have evaluated general-purpose VLMs directly on MER. [71] reports that specialized models outperform general-purpose VLMs on handwritten MER, and [72] shows that complex expressions remain challenging for general multimodal models. These results suggest that general-purpose VLM capabilities do not necessarily transfer directly to specialized MER tasks.

Evaluation is further complicated by the risk of benchmark contamination. Since the training data of many large-scale models are not fully disclosed, publicly available benchmarks may have been included in their training data, potentially leading to artificially high performance. For example, LessLeak-Bench [73] reports large performance differences between leaked and non-leaked benchmark samples.

For these reasons, only VLM-based systems providing formula-specific evaluation are considered in the following discussion. The focus remains on approaches that can be meaningfully compared with specialized PMER models.

Vision-Language Models for Document OCR and Parsing

Beyond general-purpose VLMs, recent models have been specifically developed for document parsing. These systems typically combine layout analysis, text recognition, table understanding, and MER within a unified framework. Although their primary objective is general document understanding, their formula recognition performance provides a useful reference for PMER.

Dolphin [74] splits layout understanding and recognition in two steps, while MonkeyOCR [75] and dots.ocr [76] handle document structure and content recognition at the same time. More recent systems, including MinerU2.5 [77] and PaddleOCR-VL [78, 79], use a coarse-to-fine strategy, combining global layout analysis with high-resolution recognition of local regions.

Recent systems such as DeepSeek-OCR [80], DeepSeek-OCR 2 [81], and Mistral OCR 4[3] illustrate the convergence of OCR, document parsing, and multimodal language models. Mistral OCR also highlights the issue of *polymorphic ambiguity*: different LaTeX strings can produce the same visual rendering, leading string-based metrics to count equivalent predictions as errors (Figure 2.3). This issue is addressed in Chapter 3. Equation segmentation can cause similar evaluation errors when a large expression is split into several predictions while preserving the same mathematical content.

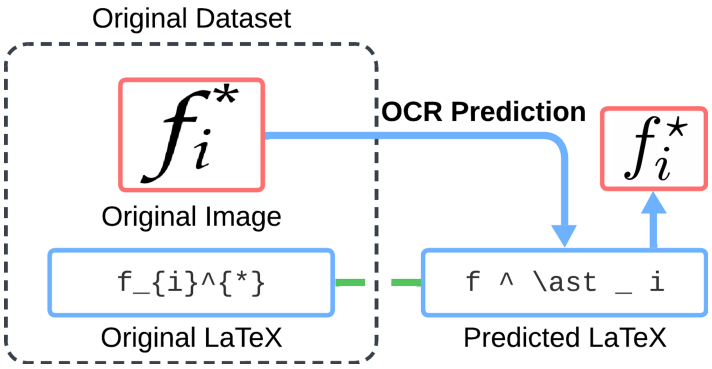


Figure 2.3: Example of polymorphic ambiguity: different LaTeX representations producing the same visual rendering.

These systems are commonly evaluated on general document benchmarks such as OmniDocBench [82], which contain heterogeneous document elements and use metrics such as FormulaCDM to assess formula recognition. While these benchmarks provide useful references for document-level systems, their setting differs from isolated PMER evaluation, where already-segmented expressions are transcribed individually. This makes their results not directly comparable with those of specialized PMER models.

Also, practical constraints limit the use of large general-purpose VLMs for the considered application. Many of the most capable models are proprietary and accessed through APIs, while large open-weight models can require a large amount of computational resources for local inference. Since the target application processes sensitive patent documents, local processing is required, making computational cost, memory requirements, privacy, and reproducibility important considerations alongside recognition accuracy.

VLM-based document parsing systems have recently achieved competitive results on general document understanding benchmarks, but their general-purpose design and heterogeneous evaluation protocols make direct comparison with specialized PMER models difficult. This thesis will focus primarily on specialized PMER approaches, while evaluating a limited selection of VLM-based systems as external baselines.

[3] https://mistral.ai/news/ocr-4/, accessed 2026-07-30.

2.2 Mathematical Expression Datasets

To train and evaluate encoder–decoder models for MER, a large amount of annotated data is required. This section provides an overview of the main datasets proposed for the image-to-LaTeX task. Although several datasets have been introduced for mathematical expression detection or content extraction from scientific documents, detection is outside the scope of our work.

The literature on mathematical expression datasets is divided between handwritten and printed expressions. This thesis focuses on printed mathematical expressions, even though handwritten datasets have played an important role in the development of the domain. The CROHME competition [60], introduced in 2011 [83], is today the main benchmark for handwritten MER. The latest editions combine human handwritten samples and synthetic data, totaling more than 160k expressions. CROHME provides both online data, through stroke trajectories, and offline data, through rendered images.

Evaluation is performed with LgEval [84], a graph-based tool that converts LaTeX or MathML representations into Stroke and Symbol Label Graphs to measure structural differences at both stroke and expression levels. More recently, the MathWriting dataset [85] introduced a large-scale corpus containing 230k human handwritten expressions and 400k synthetic expressions, collected between 2016 and 2019. Around 95% of the expressions were also sourced from Wikipedia.

For printed expression recognition, datasets consist primarily of pairs matching a visual representation of an expression with its corresponding markup representation. However, several markup formats can be used to represent the same mathematical content, and some datasets provide additional information beyond the image and markup. Those representation choices directly affect how MER systems can be trained, compared, and evaluated.

The following sections review the evolution and diversity of datasets and their implications for training and evaluating PMER.

2.2.1 Diversity of Mathematical Expression Modalities

Mathematical expressions can be represented using a wide variety of digital modalities. This section focuses on the main computer-readable formats used to encode mathematical content, ranging from textual notations and markup languages to semantic representations, spoken descriptions, and structural graphs.

For readers interested in the broader history of mathematical representation before conventional computers, [86] describes the principles and conventions used to print mathematical characters, and [87] provides an extensive historical overview of mathematical notation from ancient Babylon to the nineteenth century.

The digital communication of mathematical information requires representations capable of preserving different aspects of mathematical expressions. These aspects can be broadly divided into four levels. The *syntax* describes how an expression is written according to the rules of a representation language. The *structure* describes the relationships between the different components of the expression, such as the numerator and denominator of a fraction or the arguments of a function. The *semantics* describes the mathematical meaning of these components and their relationships. Finally, the *visual representation* describes how the expression is rendered for human readers. Several computer-readable representations have been developed to encode one or more of these aspects.

Textual Notations and Markup Languages

One of the most used representations of mathematical expressions is LaTeX [88]. It provides many commands for writing mathematical expressions, but mostly describes how they should be typeset rather than describes their mathematical semantics. The same symbol can represent different objects or roles within an expression. For example, in

$$\text{limit} = \sum_{i=1}^{n} a_i,$$

the symbol i appears four times in the raw LaTeX, but has different roles: twice as a letter in *limit*, once as the summation index, and once as the subscript of a_i.

Another mathematical markup language is MathML[4]. It was developed by the World Wide Web Consortium (W3C) in the 1990s and provides a standardized representation of mathematics on the Web.

Unlike LaTeX, MathML can represent both the presentation and the meaning of mathematical expressions. It distinguishes between *presentation markup*, which describes how an expression is displayed, and *content markup*, which describes its mathematical meaning. For example, Content MathML can represent a vector using a semantic element such as `<ci type="vector">V</ci>`, which can then be rendered according to the conventions of a given context, such as $\mathbf{V}$ or $\vec{V}$. In practice, Presentation MathML is frequently encountered in online mathematical documents.

MathML was proposed to be an important markup for communicating mathematics online [89], but its complexity and representation ambiguities have limited its adoption [90]. Tools such as the open-source API of [91] were developed to simplify MathML manipulation and adoption. A successful MathML-based resource is the Digital Library of Mathematical Functions (DLMF), a large online collection of mathematical functions released in May 2010 [92]. It makes extensive use of MathML for Web representation,

[4] https://www.w3.org/Math/whatIsMathML.html, accessed 2026-07-30.

although relatively little work has used it as a dataset. One example is [93], which provides mathematical expressions from the DLMF in LaTeX together with semantic links between corresponding pages.

Simpler textual notations and character based representations have also been developed. AsciiMath[5] provides a simple syntax that can be converted to MathML, while OpenMath aims to represent mathematical semantics in a machine-readable form [94, 95]. Mathematical expressions can also be represented directly using Unicode characters, including dedicated mathematical symbols. However, these representations do not all provide the same level of structural and semantic information. In particular, Unicode characters alone do not describe the structure of a mathematical expression.

Mathematical expressions can also be represented directly using Unicode characters rather than a markup language. Unicode includes a dedicated Mathematical Operators block[6], containing symbols such as $\int$, $\geq$, and $\in$. Greek letters and other mathematical characters are available in other Unicode blocks. While Unicode allows many mathematical expressions to be represented directly as text, Unicode characters alone do not encode their mathematical structure or semantics. It remains limited when mathematical content must be interpreted rather than simply displayed.

Natural Language and Spoken Mathematical Expressions

Mathematical expressions can also be represented using natural language. [96] introduced Formula2Text, a dataset and task for transcribing images of mathematical expressions into complete textual descriptions. The dataset contains 721 mathematical expressions associated with their LaTeX markup and five full-sentence transcriptions. For example, the expression $\beta = -\frac{1}{2\pi}R$ can be described as "beta is equal to the negative one over two pi times R". MathBridge [97] collected 23 million mathematical expressions from arXiv and open-source textbooks and converted their LaTeX representations into written spoken mathematical descriptions using GPT-3.5.

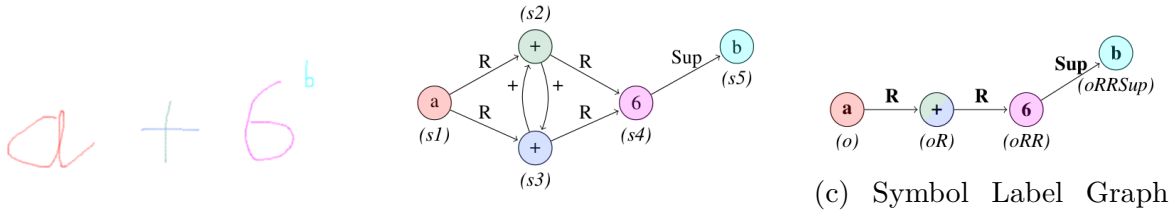
More recent work has explored the conversion of mathematical expressions between spoken language and formal mathematical notation. [98] combines spoken mathematical expressions with online handwritten mathematical expressions and MathML. The dataset contains 4,350 handwritten mathematical expressions spoken in French. The original expressions were collected from Wikipedia, and the authors also generated a synthetic subset, called *Calculator*, with a more limited vocabulary designed to represent simpler mathematical expressions. These resources illustrate that mathematical content can also be represented through natural language, although such representations are less used in PMER than image–markup pairs.

[5] https://www1.chapman.edu/~jipsen/mathml/asciimathsyntax.html, accessed 2026-07-30.
[6] https://www.compart.com/en/unicode/block/U+2200, accessed 2026-07-30.

Structural Graph Representations

Mathematical expressions can also be represented using graphs that explicitly describe their structural organization. [84] introduces *primitive Label Graphs* for the evaluation of structural pattern recognition systems. In the context of online handwritten mathematical expressions, the primitives are strokes, and the resulting representation can be viewed as a stroke-level Label Graph. They encode both the labels assigned to individual primitives and the relationships between them, making it possible to characterize classification, segmentation, and parsing errors separately or jointly through graph-based measures.

Building on this primitive-level representation, [99] introduces the *symbol Label Graph* (symLG), which represents mathematical expressions at the symbol level rather than at the level of individual strokes (see Figure 2.4). In a symLG, nodes correspond to mathematical symbols, while edges encode their spatial and structural relationships. This change in the level of abstraction makes it possible to represent typeset mathematical expressions independently of the primitives used to generate them. The authors convert LaTeX formulas into Presentation MathML and then into Symbol Label Graphs. This resulting graph representation can be used to compare the visual structure of mathematical expressions directly at the level of symbols and their relationships, rather than indirectly through the original LaTeX strings.



(a) Handwritten expression (b) Stroke Label Graph (LG) (c) Symbol Label Graph (symLG)

Figure 2.4: Example of a handwritten mathematical expression represented as a Label Graph and a Symbol Label Graph. Different colors indicate different pen strokes. Adapted from [99].

Summary of Mathematical Expression Modalities

The representations described above encode mathematical expressions at several complementary levels. An expression can be represented as a rendered image, Unicode characters, a LaTeX or MathML string, a semantic representation, a natural language description, an audio signal, or a structural graph. Each modality preserves different information and supports different applications. Figure 2.5 summarizes the modalities discussed in this section using $a + 6^b$ as a running example.

This diversity is particularly relevant to PMER, where the input is typically an image and the output may be a LaTeX sequence, a MathML representation, a semantic

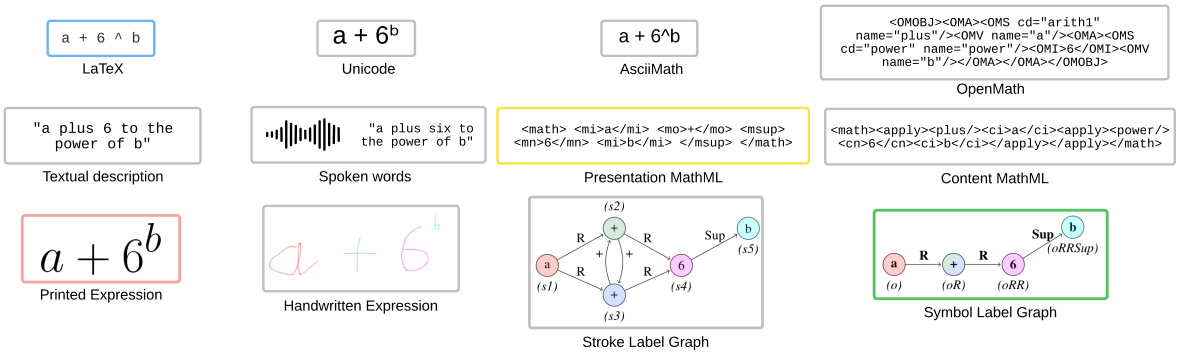


Figure 2.5: Summary of different mathematical expression modalities for the expression $a + 6^b$. Stroke and Symbol Label Graphs representations are adapted from [99].

structure, or another machine-readable format. However, most existing datasets focus on image–LaTeX pairs. The choice of output representation directly affects both dataset annotation and the way MER systems can be evaluated. The following sections mainly consider datasets providing this type of annotation.

2.2.2 Early datasets

Early datasets for MER mainly consist of scanned scientific documents. They are valuable for studying robustness to real-world scanning conditions, but are generally small and difficult to access. Studies such as [20, 100] reviewed the challenges of MER for large collections of scanned scientific documents. They highlighted scanning noise, binarization artifacts, touching characters, font and size variations, and the two-dimensional structure of mathematical expressions.

One of the earliest examples of dataset is the UW document image database [101]. The UW-I, UW-II, and UW-III databases contain over two thousand English and Japanese technical document images. In UW-III, 25 pages containing mathematical formulae were selected and their mathematical zones annotated, including formula locations and individual symbols. InftyCDB-1 [102] contains characters and symbols from 467 pages of 30 mathematical articles, annotated with attributes such as character type, font, image quality, and position.

These early datasets established the importance of realistic variations in scanning, typography, and document layout for MER. However, their limited scale, accessibility, and annotation formats made them unsuitable for training modern data-hungry recognition models.

2.2.3 Large-Scale Synthetic Datasets from LaTeX Sources

A large proportion of modern printed mathematical expression datasets follows a synthetic generation pipeline in which mathematical expressions are extracted from

scientific documents as LaTeX source markup and rendered as images. This approach makes it possible to generate large-scale image–formula pairs at relatively low cost. The resulting images generally lack the noise, distortions, and variability associated with real-world document acquisition, such as scanning artifacts, compression, blur, and degradation.

Table 2.3 summarizes the datasets presented in this subsection, with their source, content, visual characteristics, size, and main contribution.

The im2latex100k Dataset and Its Successors

The most influential dataset in this line of work is im2latex100k [22]. It was constructed from the KDD Cup 2003 dataset [107], which contains approximately 225,000 physics articles from arXiv. The authors extracted 103,556 displayed mathematical expressions and rendered them as images, as illustrated in Figure 2.6. im2latex100k became a widely used benchmark for PMER. The extracted LaTeX was processed using a KaTeX-based parser before rendering to reduce inconsistencies between visually equivalent expressions.

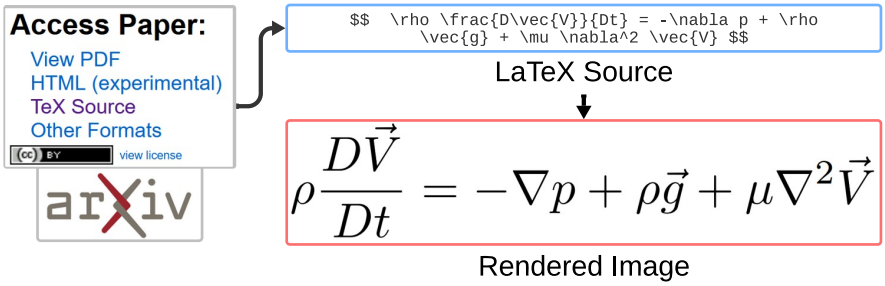


Figure 2.6: Illustration of the pipeline used to generate paired mathematical expression images and LaTeX from arXiv articles.

Several datasets reused the same source and acquisition strategy. im2latex90k and im2latex140k [27] filtered some very large samples from im2latex100k and added expressions from more arXiv papers. ME98k [52] also provided a cleaner subset by removing some examples. Other works developed tools to parse the original KDD Cup sources and generate new image–LaTeX pairs [108], leading to other redistributions such as im2latex170k [103] and im2latex230k [104]. This creates a risk of overlap between these collections and can result in duplicated or near-duplicated expressions.

The IBEM dataset [105] also uses the KDD Cup source but extends it to both displayed and inline expressions, with several LaTeX preprocessing levels. Unlike the isolated formulas that dominate traditional MER benchmarks, inline expressions occur within surrounding text and are generally shorter, providing a different data distribution.

Dataset	Source	Content	Visual Characteristics	Size	Main Contribution
Foundational Datasets and Their Successors					
im2latex100k [22]	Scientific papers (arXiv)	Displayed mathematical expressions	Clean synthetic rendering	103k	Foundational large-scale benchmark for image-to-LaTeX recognition
im2latex90k/140k [27]	Scientific papers (arXiv)	Displayed mathematical expressions	Clean synthetic rendering	90k–140k	Filtering of problematic samples and expansion of the original dataset
ME98k [52]	Scientific papers (arXiv)	Displayed mathematical expressions	Clean synthetic rendering	98k	Cleaner subset with reduced annotation and rendering inconsistencies
im2latex170k/230k [103, 104]	Scientific papers (arXiv)	Displayed mathematical expressions	Clean synthetic rendering	170k–230k	Large-scale redistributions and extensions of the original source
IBEM [105]	Scientific papers (arXiv)	Displayed and inline mathematical expressions	Clean synthetic rendering	166k	Introduces a large collection of inline expressions in addition to displayed formulas
Alternative Sources and Visual Diversity					
im2latexv2 [58]	Scientific papers (arXiv)	Displayed mathematical expressions	59 fonts, 600 DPI rendering	100k	Increases typography and rendering diversity while preserving the original formulas
realFormula [58]	Scientific papers (arXiv)	Manually annotated mathematical expressions	Clean synthetic rendering	121	Evaluates recognition quickly on manually annotated formulas

Table 2.3: Summary of datasets for PMER. The datasets are grouped according to their source, mathematical content, and visual characteristics.

Dataset	Source	Content	Visual Characteristics	Size	Main Contribution
Alternative Sources and Visual Diversity (continued)					
ME20k [52, 29]	Online educational platform	Expressions from mathematical exercises	Different source distribution and visual styles	20k	Provides a non-scientific-paper distribution for evaluating domain transfer
AMME-20K	Scientific papers	Multi-line mathematical expressions	Real document images	20k	Dedicated benchmark for multi-line mathematical expression recognition
Fusion Image-to-LaTeX [106]	Multiple existing datasets	Printed and handwritten mathematical expressions	Heterogeneous	3.4M	Combines multiple datasets and applies LaTeX normalization
Tex80M	Scientific papers (arXiv)	Synthetic mathematical expressions	Multiple fonts, textures, noise, lighting and degradation	80M+	Large-scale synthetic pretraining with increased visual and structural diversity
HDR [55]	Scientific papers (arXiv)	Hierarchically complex mathematical expressions	Synthetic rendering	100M+	Enables evaluation according to structural complexity, formula length and number of lines
UniMER [59]	Scientific papers, online encyclopedias and real documents	Simple, complex, screen-captured and handwritten mathematical expressions	Fonts, backgrounds, distortions and image quality variations	1M+	Broad benchmark combining structural complexity and realistic visual variability

Table 2.3: Summary of datasets for PMER (continued). These datasets focus on scaling synthetic training data, increasing visual diversity, and evaluating recognition under complex or realistic conditions.

Increasing Visual Diversity and Realism

ME20k [52, 29] provides a different source distribution from the scientific-paper-based datasets discussed above. It was collected from mathematical exercises published on the Chinese online education platform Zhixue.com, making it particularly useful for evaluating models trained on other domains and testing transfer learning.

A further direction has been to increase visual diversity rather than only the number of formulas. The im2latexv2 dataset, introduced with MathNet [58], re-renders im2latex100k expressions using 59 mathematical fonts at 600 DPI. This increases variation in typography and rendering conditions while preserving the original mathematical content. The same work introduced realFormula, a small and manually annotated test set extracted arXiv documents and rendered as images.

Tex80M, introduced with TexTeller [57], scales this approach to more than 80 million synthetic formula images generated from formulas extracted from approximately 300,000 arXiv papers. The authors use a unified extraction pipeline based on Nougat [46] to normalize LaTeX expressions and remove author-specific macros and styles. The formulas are then augmented through concatenation, operator and bracket transformations, diverse fonts, paper textures, lighting variations, noise, and ink degradation. Tex80M demonstrates the potential of large-scale synthetic data for improving recognition performance and generalization.

Fusion Image-to-LaTeX [106] follows a similar strategy at a smaller scale by combining approximately 3.4 million image–text pairs from several existing datasets, including im2latex100k, im2latex90k, im2latex140k, im2latex170k, im2latex230k, CROHME, and the Aida Calculus Math Handwriting Recognition Dataset [109]. It contains 3.2 million printed and more than 200,000 handwritten expressions. LaTeX normalization using a KaTeX parser reduces polymorphic ambiguity, although combining datasets derived from the same sources poses the risk of duplicate expressions.

Recent datasets have also targeted more complex structures. AMME-20K, introduced for the ICPR 2024 competition on multi-line MER [110], contains 20,000 expressions from real academic papers, with approximately 60% containing multiple lines. The HDR dataset [55] focuses on hierarchical complexity, from simple characters to deeply nested structures such as fractions and matrices. Its HDR-100M training set contains more than 100 million synthetic formula images, while HDR-Test contains 43,932 formulas annotated for evaluating complexity, number of lines, and formula length.

The UniMER dataset [59] was designed to address the limited diversity and complexity of earlier benchmarks. UniMER-1M contains 1,061,791 LaTeX-image pairs, while UniMER-Test contains 23,789 expressions across four categories: simple printed expressions (SPE), complex printed expressions (CPE), screen-captured expressions (SCE), and handwritten expressions (HWE). Its printed expressions come from arXiv,

Wikipedia, and StackExchange, and screen-captured expressions are taken from more than 1,000 pages of books, papers, textbooks, magazines, and newspapers in Chinese and English. The dataset covers variation in font, size, background, image quality, distortion, and writing style, as well as a broad range of expression complexity.

LaTeX-based datasets have driven the development of PMER by providing large, diverse, and complex training resources. However, the increase in dataset size has not always been accompanied by greater diversity in style or source documents, and models are still often trained on im2latex100k or closely related datasets. This creates a risk of overlap between train and test data and makes it difficult to evaluate generalization to previously unseen documents. There is also limited evaluation of real-world robustness, particularly for low-quality scans and photographs. Recent resources such as UniMER address this limitation by incorporating more diverse sources, screen-captured expressions, broader visual variability, and different levels of structural complexity.

2.2.4 Large-scale document understanding benchmarks

Recent multimodal models are increasingly evaluated on large-scale document understanding benchmarks covering tasks such as text recognition, layout analysis, table parsing, formula recognition, information extraction, and visual question answering [111]. Some also include real-world documents, such as scanned or photographed pages. However, mathematical content generally represents only a small part of these benchmarks.

General-purpose benchmarks often contain too little mathematical content to serve as dedicated PMER benchmarks. OmniDocBench [82] contains 981 document pages from nine document types and includes 4,009 inline mathematical formulas annotated in LaTeX. The formulas were initially generated using UniMERNet [59], then checked and corrected manually. Similarly, olmOCR-Bench [112] contains 1,402 PDF documents and 7,010 *test cases*, including 3,385 mathematical tests. These comprise 2,927 expressions from arXiv papers and 458 expressions from scanned historical mathematics textbooks. The benchmark evaluates whether the OCR output preserves the expected symbols and their spatial relationships rather than relying only on string-based metrics. However, mathematical recognition is evaluated as one component of a broader document OCR benchmark rather than as a dedicated PMER task.

Other benchmarks contain even fewer mathematical samples. OCRBenchV2 [113] evaluates 23 tasks using 10,000 human-validated instruction-response pairs, but its formula recognition task contains only 475 handwritten formulas. CC-OCR [114] contains 7,058 annotated images across 39 subsets, but only approximately 100 handwritten mathematical formulas and 100 chemical formulas. Such quantities are insufficient for a dedicated evaluation of PMER.

Recent benchmarks have focused on robustness to real-world document acquisition. WildDoc [115] evaluates documents captured in natural environments, DocPTBench [116] focuses on photographed documents and geometric distortions, and Real5-OmniDocBench [117] reconstructs OmniDocBench documents under five physical conditions, including scanning, warping, screen photography, illumination changes, and skew. These benchmarks show that document understanding performance decreases under physical capture conditions, but none specifically evaluates PMER under such conditions.

Proprietary OCR systems also use non-public evaluation protocols. Mistral, for example, reports results for Mistral OCR 4[7] on more than 600 documents in over 12 languages using blind human comparisons with five competing OCR systems. It also reports scores of 85.20% on olmOCR-Bench and 98% on its internal Crawl Multilingual evaluation. However, the internal documents and annotations are not publicly available, limiting reproducibility and direct comparison.

Overall, existing large-scale document benchmarks are not well suited to high-accuracy PMER evaluation. Their mathematical content is often limited, mixed with handwritten or non-mathematical content, or treated as only one task among many. Moreover, none provides a large-scale evaluation of PMER under realistic acquisition conditions. Dedicated PMER benchmarks remain necessary for evaluating specialized systems in high-accuracy settings.

2.2.5 Patent-based datasets

To the best of our knowledge, no publicly available dataset specifically targets MER from patent documents. Existing patent datasets instead focus on patent text, technical drawings, figure understanding, image retrieval, and multimodal analysis.

Several resources use data from the United States Patent and Trademark Office (USPTO). DeepPatent and DeepPatent2 [118, 119] provide more than 350,000 and 2.7 million patent drawings, respectively. PatentLLM [120] introduces PATENTDESC-355K, containing approximately 355,000 patent figures paired with textual descriptions. Other USPTO-based resources focus on figure annotations [121], patent text and metadata [122], or document summarization [123].

EPO-based datasets also target broad patent analysis. [124] uses EPO patents for image retrieval, while PatentMatch [125] provides 6.26 million claim and prior-art paragraph pairs. [126] collects 73,980 European patents through the EPO API, and PatFig [127] provides 62,513 patent images from 15,645 European patent applications linked to textual descriptions.

Other collections combine multiple patent sources. The EuroPat Corpus [128]

[7] https://mistral.ai/news/ocr-4/, accessed 2026-07-30.

provides multilingual patent text from the USPTO and EPO, while the CLEF-IP collection [129] is based mainly on the MAREC corpus[8] and contains approximately 3.5 million XML documents corresponding to 1.5 million patents.

Existing patent datasets provide many resources for text processing, technical drawing understanding, and image retrieval, but do not address MER. This constitutes a clear gap, despite the large volume of patent documents containing mathematical content.

Overall, older PME datasets provide realistic scanned-document variability but are generally small and difficult to access, whereas recent large-scale datasets still rely heavily on synthetically rendered expressions, although some works increasingly incorporate visual degradation and real-world acquisition conditions. General-purpose multimodal benchmarks contain more realistic documents but treat mathematical recognition as only one component of broader OCR tasks, while proprietary evaluations are often difficult to reproduce. Consequently, existing PME research remains heavily dependent on synthetic data, and no publicly available dataset specifically addresses MER in patents.

2.3 Evaluation Metrics for MER

Evaluating MER systems remains challenging due to the lack of standardized benchmarks and evaluation protocols. Early studies highlighted inconsistencies between evaluation methods, making fair comparisons difficult [20]. The emergence of deep learning highlighted the need for metrics that reflect both recognition accuracy and human judgment [130, 131].

MER predictions and ground-truth can be compared in several modalities, including text, rendered images, and structural graphs, but the conversion and normalization of these representations directly affect evaluation. The first subsection discusses these conversions, before reviewing text-, visual-, graph-, and embedding-based metrics.

2.3.1 Conversion and Normalization

Mathematical expressions can be converted between LaTeX, MathML, graph representations, and images (see Section 2.2.1). These conversions are useful for both data generation and evaluation, particularly when equivalent expressions have different textual or syntactic forms.

Several tools convert LaTeX to MathML, such as Tralics [132], BlahTeXML [133], and LaTeXML [134]. LaTeXML converts LaTeX documents into an intermediate XML representation that can be used to generate Presentation and Content MathML,

[8] https://ifs.tuwien.ac.at/imp/marec, accessed 2026-07-30.

HTML and EPUB. It has been used in the arXMLiv project to convert more than 90% of the arXiv collection to HTML [135]. Comparative studies have also evaluated LaTeX-to-XML converters in terms of coverage and MathML quality [136].

For image generation, LaTeX can be rendered using any TeX engine, while MathML can be rendered by most browsers. Generative approaches have also been proposed for converting mathematical markup into images, but they are less suitable when deterministic and reproducible rendering is required [137].

Normalization is important because mathematically equivalent expressions may have different textual or structural representations. Previous work has addressed this through MathML canonicalization [138] and parse-tree normalization [139]. Similar approaches are used for LaTeX: Tex80M [57] uses Nougat [46] to reduce variations caused by author-defined macros, while Deng et al. [22] normalize extracted LaTeX using KaTeX.

Finally, expressions can be represented as structural graphs. The symLG representation converts normalized MathML into graphs whose nodes represent symbols and whose edges encode their relationships, enabling fine-grained structural comparison beyond purely textual or visual metrics [99].

The conversion and normalization steps are not neutral preprocessing operations. Their design can affect the equivalence classes of expressions, the resulting annotations, and the evaluation scores. A robust evaluation protocol must make the conversion and normalization procedures explicit and reproducible.

The choice of representation also directly affects how predictions are compared with references. The following sections consider several families of metrics, which evaluate agreement between predictions and references from different perspectives, including textual, visual, structural, and embedding-based representations.

2.3.2 Text-based metrics

Exact Match is one of the most common metrics for MER. It measures the proportion of predictions that are identical to the ground-truth. However, it is highly sensitive to representation differences: two different LaTeX sequences can produce the same visual expression while receiving a score of zero, as illustrated previously in Figure 2.3.

Levenshtein distance [140] provides a less strict alternative by measuring the minimum number of insertions, deletions, and substitutions required to transform the prediction into the reference. BLEU [25] measures (n)-gram precision with a brevity penalty. These metrics provide more gradual scores than Exact Match but remain dependent on the textual representation and do not explicitly measure mathematical equivalence. They remain limited when different representations correspond to the same mathematical expression, motivating visual and structural alternatives.

2.3.3 Visual metrics

Visual metrics compare rendered images of predicted and reference expressions, reducing the effect of polymorphic LaTeX representations. However, they can be sensitive to small rendering differences, such as misalignment or changes in image dimensions.

Simple approaches include pixel-wise comparison, image correlation [141], and the Structural Similarity Index Measure (SSIM) [142], which compares luminance, contrast, and structural information. However, such metrics may penalize small visual differences even when the recognized expression is largely correct.

Several metrics have been specifically developed for MER. Zanibbi et al. [143] compare expressions using connected-component features such as contours, pixel density, aspect ratio, and spatial layout. IMEGE [144] uses a Binary Image Distortion Model to compare corresponding image regions while allowing local distortions. It computes precision- and recall-like scores and combines them using their harmonic mean. The image-based Levenshtein distance [26] converts vertical image slices into discrete sequences and applies Levenshtein distance to the resulting representations. This provides an image-level edit distance that is less dependent on the original LaTeX representation.

More recently, Character Detection Matching (CDM) [145] renders the predicted and reference formulas, detects and matches their individual elements using graph matching, and considers token identity, spatial proximity, and order. The final score is an F_1-score over correctly matched elements.

Visual metrics are less dependent on the textual form of the prediction, but they remain dependent on the rendering process and do not directly encode the mathematical structure of the expression. They provide a useful complementary view of recognition quality, particularly when alternative LaTeX representations produce visually equivalent results.

2.3.4 Graph-based metrics

Graphs provide another powerful representation for mathematical expressions. Once an expression is represented as a graph or tree, a wide range of graph-comparison metrics can be applied, including graph edit distance, tree edit distance, and graph matching.

For handwritten MER, LgEval [84] attempts to mitigate this issue for handwritten MER evaluation by employing a graph-based representation and Hamming distance to quantify differences between expressions. It distinguishes three types of errors: classification errors ΔC (symbol recognition), segmentation errors ΔS (stroke merging), and relationship errors ΔL (relations between symbols). These errors are averaged into the $\Delta E(\%)$ metric, defined as $\Delta E(\%) = \frac{1}{3}\left(\frac{\Delta C}{n} + \sqrt{\frac{\Delta S}{n(n-1)}} + \sqrt{\frac{\Delta L}{n(n-1)}}\right)$. The Symbol Label Graph library [99] extends this approach by converting MathML expressions into

Label Graph format, allowing the evaluation of PMER with this tool.

Graph- and tree-based evaluation has also been applied to mathematical expressions, which are naturally represented as XML trees in MathML. EMERS [146] converts MathML expressions into trees and evaluates recognition performance using tree matching and the distance between the corresponding Euler strings. Similarly, other approaches construct tree representations from MathML and compare expressions using tree-matching techniques or normalized structural representations [147, 148]. These methods can capture structural differences that are difficult to represent using purely textual metrics.

Overall, graph and tree-based methods capture structural differences that are difficult to represent with textual or visual metrics, but their effectiveness depends on the underlying representation and comparison method.

2.3.5 Embeddings

Embedding-based metrics compare predictions and references in a continuous vector space, allowing similarities beyond direct textual or visual matching. General evaluation methods such as BERTScore [149] and COMET [150] rely on learned representations rather than predefined token overlap. These approaches could potentially be adapted to mathematical expressions using models such as MathBERT [151]. TeXBLEU [152] applies a similar idea specifically to LaTeX using a specialized tokenizer and embedding-based similarity.

Several works in Mathematical Information Retrieval (MIR) have explored mathematical embeddings. Formula2Vec [153] learns dense representations of LaTeX formulas and compares them using cosine similarity, while Equation Embeddings [154] represents equations through their surrounding textual context to capture semantic meaning. Other approaches incorporate both formula and document information, such as the TF-IDF and Doc2Vec representations used in [155].

Embedding methods have also been learned directly from visual or structural representations. [156] learns embeddings from formula images using convolutional neural networks, while [157] represents formulas as operator trees and learns embeddings with a tree-based autoencoder. EARN [158] combines a visual encoder with a graph neural network to learn an embedding space reflecting both visual and structural similarity.

These approaches provide alternatives to exact string matching by capturing semantic, visual, or structural similarity. However, their direct use for MER evaluation is challenging, because OCR evaluation often requires distinguishing very small structural differences that can be missed by embedding architectures. They are particularly promising for applications such as retrieval, clustering, or quality assessment, but are less suitable as metrics for high-precision MER evaluation.

Overall, current evaluations often rely on a single modality. A more comprehensive evaluation could combine textual (LaTeX), structural and semantic (MathML, LgEval), and visual metrics to provide a more robust assessment of MER systems. In particular, the potential of MathML-based evaluation remains largely underexplored.

2.4 Discussion and Conclusion on the State of the Art for MER

This state of the art reviewed MER from three complementary perspectives: recognition models, datasets, and evaluation metrics.

Regarding recognition models, the literature shows a transition from traditional pipeline-based methods, which separated symbol recognition from structural analysis, to end-to-end deep learning. Vision Transformer-based encoder–decoder architectures have become a dominant family of specialized PMER models due to their strong recognition performance and ability to jointly model visual and structural information. Although general-purpose Vision-Language Models perform well on broader document understanding tasks, specialized MER models remain better suited to high-accuracy mathematical recognition.

Datasets have evolved from small collections of scanned documents with realistic noise and scanning artifacts, to large-scale synthetic datasets generated from LaTeX. These datasets provide the scale required by modern deep learning models but they often lack visual diversity and realism. Some recent work has introduced variations in fonts, backgrounds, rendering quality, and image degradations. General document understanding benchmarks provide some realistic conditions, but mathematical expressions usually represent only a small part of their content. Also, they mostly use LaTeX as the target representation, while structurally richer alternatives such as MathML are underused.

For evaluation, text-based metrics such as Exact Match and edit distance measure sequence accuracy but are sensitive to the markup language. Visual and graph-based metrics provide more information by capturing visual and structural similarity, and embedding-based approaches offer a learned way to measure similarity. However, each family of metrics captures only part of recognition quality, and their results depend on the representation, conversion, and normalization procedures that have been used. No single metric fully captures the different dimensions of MER quality.

The literature highlights three main challenges for reliable PMER:

- **Data diversity and independence:** recognition performance depends not only on model architecture, but also on the diversity and independence of the data used for training and evaluation.
- **Evaluation reliability:** existing evaluation protocols do not always capture the structure of mathematical expressions correctly, especially when different representations produce the same visual result or when small structural errors change the meaning of an expression.
- **Real-world reliability:** the gap between benchmark performance and correct recognition in real processing conditions remains difficult to measure, particularly when very high precision is required.

These observations motivate the work presented in the following chapters, which addresses PMER recognition, data diversity, robust evaluation, and controlled automation.

Chapter 3

PiE-MER: a Robust Multimodal Mathematical Expression Recognition Evaluation Pipeline

This chapter is based on results reported in [159].

3.1 Introduction

The previous chapter presented the state of the art in MER, covering model architectures, datasets, evaluation metrics, and comparison protocols. Despite the progress made in MER with recent approaches based on Vision Transformers, the evaluation of recognition systems is heterogeneous. Existing studies often rely on different datasets, representations, and evaluation protocols, making direct comparisons difficult.

This chapter focuses on the evaluation of Offline MER, where the input is a static image of a mathematical expression. The main contribution of this chapter is **PiE-MER**, a multimodal ***Pi****peline for the* ***E****valuation of* ***M****athematical* ***E****xpression* ***R****ecognition*. In this work, multimodal evaluation means using multiple representations obtained from the same mathematical expression to measure performance from different perspectives. PiE-MER converts predicted LaTeX into MathML, Symbol Label Graphs [99], and normalized images to evaluate the prediction at different levels while accounting for **polymorphic ambiguity**. The framework is evaluated on five MER models and eight datasets, covering different architectures and generations of recognition systems.

This chapter presents the proposed evaluation framework and its application to existing MER systems. Section 3.3 describes the methodology of the evaluation pipeline, including the multimodal representations and evaluation metrics. Section 3.4 presents the evaluation results. Section 3.5 analyzes the results and discusses the main findings and recommendations for future research. Finally, Section 3.6 summarizes the chapter and discusses future improvements for MER evaluation.

3.2 Motivation

Traditional MER evaluation often relies on sequence-based metrics such as Levenshtein distance and BLEU applied to LaTeX markup. These metrics measure string similarity and may fail to measure structural or visual equivalence. A prediction can be considered incorrect on the LaTeX level while producing the same rendered expression as the reference. For example, the LaTeX expressions `x^{2}\_{i}` and `x\_i^2` both render as x_i^2, despite being different on the LaTeX level. This **polymorphic ambiguity** makes exact string matching insufficient for reliably evaluating LaTeX predictions. This issue has also been highlighted by the authors of Mistral OCR 4[1], who report that different LaTeX expressions rendering identically can be counted as mismatches.

The problem is particularly relevant in the context of technical document processing and retrieval, where a very high level of accuracy is required. An evaluation protocol should distinguish between syntactic differences that do not affect the resulting expression and genuine recognition errors.

[1] https://mistral.ai/news/ocr-4/, accessed 2026-07-30

The heterogeneity of existing evaluation protocols motivates the need for a unified framework. Differences in datasets, output representations and metrics can lead to different conclusions about the performance of MER systems. A multimodal evaluation protocol is needed to determine whether conclusions drawn from standard sequence-based metrics remain valid when structural and visual equivalence are also considered. PiE-MER addresses these limitations by combining multiple representations for its evaluation. It compares LaTeX predictions and ground-truth by converting them into MathML, Label Graph, and normalized image representations, and should provide a broader and less biased assessment of recognition quality.

3.3 Methodology

3.3.1 Selected Models for Evaluation

To study the evolution of architectures for PMER, five models were selected for evaluation. The selection was based on three main criteria: reported performance, architectural diversity, and the availability of pretrained weights and implementations[2]. The selected models cover the main architectures used in recent MER systems, including CNN–Transformer architectures, Vision Transformer-based models, and encoder–decoder architectures derived from general image-to-sequence recognition systems. The evaluation focuses on models developed between 2016 and 2024, providing a representative sample of the evolution of MER architectures while keeping the benchmark computationally manageable.

Table 3.1 summarizes the key details of the selected models. We aim to investigate how architectural differences, the amount of training data and its diversity, and input resolution determines recognition performance and generalization across datasets. Since the training datasets used for these models vary significantly, and the details of their test sets are not always fully specified, we re-evaluate them all under consistent conditions on the available test splits of multiple datasets for a more consistent comparison. We use the models' published weights without any fine-tuning to measure their generalization capabilities.

We call the first selected model **KYS**. It was proposed in 2021 in the paper *Full Page Handwriting Recognition via Image to Sequence Extraction* [160], and adapted to Printed MER by GitHub user "kingyiusuen". It uses a simple encoder–decoder architecture, with a ResNet-18 encoder with two-dimensional positional encoding and a Transformer decoder. The model is trained on the im2latex100k dataset. Its relatively simple

[2]Repositories available at: github.com/kingyiusuen/image-to-latex; github.com/lukas-blecher/LaTeX-OCR; github.com/VikParuchuri/texify; github.com/OleehyO/TexTeller; github.com/felix-schmitt/MathNet. Accessed 2026-08-22.

Table 3.1: Overview of MER models, their architectures, training set, and key metrics.

Model	Year	DPI	Architecture (Encoder → Decoder)	Training Set	Edit Score	BLEU
KYS	2021	200	ResNet-18 (2DPE) → Transformer	im2latex100k	83.00	–
pix2tex	2021	100	CNN + ViT → Transformer	im2latex100k, Wikipedia, arXiv	90.00	88.00
Texify	2023	100	Swin ViT → mBART	im2latex100k, arXiv	93.50	84.20
TexTeller 2.0	2024	100	ViT → Transformer	7.5M formulas	–	–
MathNet	2024	600	3 layer CvT → Transformer	im2latexv2	97.2	96.8

CNN encoder–Transformer decoder design makes it a great baseline for comparing conventional image-to-sequence architectures with more recent and larger models.

The second selected model is **pix2tex**, an image-to-sequence architecture based on a Vision Transformer encoder and a Transformer decoder. It was developed for MER and trained on a combination of the im2latex100k dataset, Wikipedia, and arXiv data. Its Vision Transformer-based architecture makes it a representative of early Transformer-based approaches to MER.

The third selected model is **Texify**. Texify uses a Vision Transformer-based encoder inspired by Donut and a Transformer-based mBART decoder. Unlike models designed exclusively for isolated mathematical expressions, it is intended to process images containing both text and mathematical content. Its variable-input preprocessing pipeline preserves the aspect ratio of the input image before padding it to a fixed input size. Texify was selected as a representative of more recent Vision Transformer-based approaches and because of its adaptive preprocessing strategy.

The fourth model is **TexTeller 2.0**. This model uses a relatively simple Vision Transformer–Transformer encoder–decoder architecture, but benefits from training on a larger-scale synthetic dataset than the earlier im2latex100k-based systems. In particular, TexTeller 2.0 is trained on approximately 7.5 million mathematical expressions, illustrating the increasing importance of training-data scale in MER. A newer version of the model, TexTeller 3.0, has since been released and trained on the much larger Tex80M dataset. However, TexTeller 2.0 is still relevant for the present preliminary benchmark as the objective of this study is to evaluate the proposed evaluation methodology on representative architectures.

Finally, **MathNet** was selected as a representative hybrid CNN–Transformer architecture. Its encoder alternates convolutional and Transformer blocks before generating the output through a Transformer decoder. The model is trained on im2latexv2, a multi-font extension of the original im2latex100k dataset in which the mathematical expressions are rendered using 59 different fonts. This design makes MathNet particularly relevant for evaluating the robustness of MER systems to variations in visual styles. In addition, the model is particularly sensitive to the resolution of the input image. The authors report their best performance when expressions are natively rendered

at a resolution of 600 DPI, with lower-resolution inputs resulting in a degradation of recognition performance. These two characteristics make MathNet especially relevant to the present study, which investigates both the impact of input resolution and the generalization of MER systems across visually diverse mathematical expressions.

These five models differ in their architecture, number of parameters, training data, training resolution, and preprocessing strategies. They cover a progression from relatively compact CNN–Transformer encoder–decoder architectures to large-scale Vision Transformer-based systems trained on millions of synthetic mathematical expressions. This diversity makes it possible to study whether the proposed evaluation methodology remains consistent across different generations and architectural families of MER systems.

The selected models also differ in their required input image resolution. Some can recognize images rendered at any resolution, whereas others, such as MathNet, report their best results with expressions rendered natively at 600 DPI. Several models apply model-specific preprocessing operations, including automatic resizing, aspect-ratio preservation, and padding. Applying a generic resizing strategy to all models could therefore introduce an evaluation bias and make the comparison dependent on preprocessing choices rather than recognition capability. That is why we designed a custom image rescaling module, illustrated in Figure 3.1. For each model, the input image can either be processed at its original resolution or rescaled according to the resolution recommended by the corresponding model. Each model is evaluated under these two conditions. This makes it possible to quantify the influence of input resolution on recognition performance and to determine whether the reported performance of individual MER systems depends on the resolution of the input data. The results of this analysis are presented in Section 3.4.

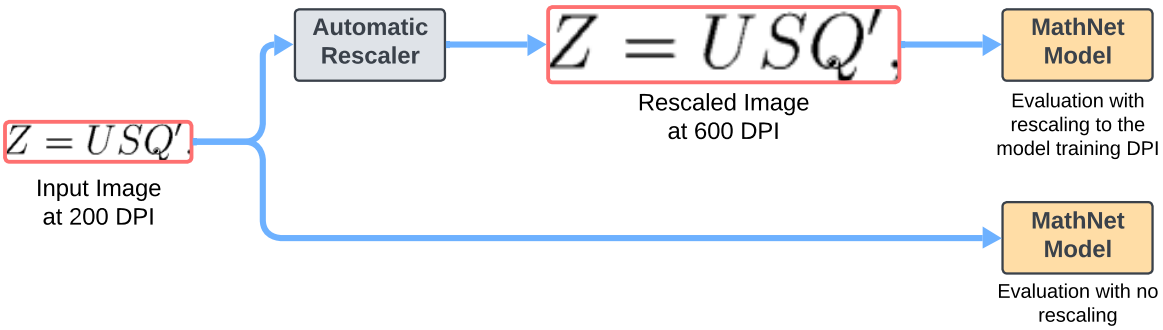


Figure 3.1: Illustration of the proposed image rescaling process. Each model is evaluated both with the original input images and with images rescaled according to the resolution recommended for the corresponding model. The example shows the rescaling used for MathNet, which was trained with images rendered at 600 DPI.

3.3.2 Benchmark Datasets for Evaluation

We selected eight publicly available benchmark datasets covering complementary properties of printed mathematical expressions. This diversity allows us to measure whether our multimodal evaluation framework remains reliable across different expression distributions, styles, and annotation conventions. The selected datasets are summarized in Table 3.2.

Table 3.2: Overview of selected datasets, with their release year, resolution in DPI, total sample count, test set size before and after filtering, and description.

Name	Year	DPI	Samples	Test split	Filtered Samples	Description
i2l100k-raw	2016	200	103,556	10,325	10,325	Raw LaTeX from arXiv papers
i2l100k-parsed	2016	200	103,556	10,283	10,283	Normalized LaTeX expressions
i2l90k	2018	200	93,741	4,648	4,647	Cleaned i2l100k subset
i2l140k	2018	200	142,966	14,280	14,279	Superset of i2l90k
ME20k	2020	100	20,834	2,083	2,083	High-school level math from Zhixue.com
IBEM	2023	200	166,692	44,383	44,367	Includes inline and display expressions
i2lv2	2024	600	92,606	10,118	10,077	i2l100k with 59 font variations
realFormula	2024	600	121	121	121	Manually annotated

Most datasets originate from the **im2latex100k** benchmark and its successors, which remain the standard for evaluating PMER. We included two versions of the original dataset in our benchmark: **100k-raw** contains the original LaTeX expressions extracted from arXiv papers, whereas **100k-parsed** uses the KaTeX-based parser introduced by Deng et al. in [22] to reduce ambiguities caused by visually equivalent LaTeX representations. The datasets **im2latex90k** and **im2latex140k** [27] filter some expressions by removing samples that fail to render correctly or have excessively large image dimensions. Evaluating these datasets allows us to quantify the impact of dataset curation and filtering on evaluation results.

IBEM [105] explicitly includes both displayed and inline mathematical expressions, with approximately 80% of its samples corresponding to inline formulas. Since inline expressions are typically shorter and contain a different symbol distribution from displayed equations, they constitute an interesting evaluation scenario. We used the recommended *latex_expand* version of the IBEM annotations.

To extend the evaluation beyond the classical LaTeX rendering pipeline, we also include a dataset introducing additional visual diversity. **im2latexv2** [58] re-renders the im2latex100k mathematical expressions using 59 different mathematical fonts at 600 DPI. This dataset allows us to evaluate the impact of font styles on model performance while highlighting the relevance of our multimodal evaluation framework, which renders mathematical expressions using a standardized process to minimize the influence of

visual differences in the source images. We also included **realFormula**, a small dataset released alongside im2latexv2. It contains only 121 manually annotated expressions extracted by the authors from real scientific documents, making it useful for quickly validating and debugging the evaluation pipeline.

Finally, the benchmark also includes a dataset of expressions that were not collected from arXiv. **ME20k** [52] contains mathematical expressions extracted from exercises on the zhixue.com online educational platform. Its expressions are shorter and generally simpler than those found in scientific publications.

For consistency, we used the datasets' tokenized expressions to build a shared vocabulary of 601 LaTeX tokens. It is used to perform token-level metric computation. We use this vocabulary to re-tokenize the raw string outputs of the different models. We also filtered out erroneous expressions from the datasets, such as those with empty LaTeX annotations, slightly reducing the size of some test sets.

Taken together, these datasets provide a diverse evaluation benchmark in terms of expression sources, rendering conditions and expression complexity. This diversity allows us to investigate whether the choice of dataset affects the conclusions drawn about MER system performance.

3.3.3 Evaluation Pipeline and Metric Selection

For each evaluation of a model on a given dataset, the pipeline starts by rescaling the input images according to the DPI requirements of the evaluated model. The images are then provided to the model to generate LaTeX predictions. Since all MER models considered in this work generate LaTeX outputs, this representation is used as the starting point of our conversion pipeline. The predicted LaTeX expressions, as well as the ground-truth annotations, are then converted into three additional modalities: MathML, Label Graph, and normalized rendered images.

The complete conversion pipeline is illustrated in Figure 3.2. Starting from an original dataset sample composed of an image and a LaTeX annotation, the pipeline generates the three additional modalities. The complete evaluation process, applied identically to both ground-truth expressions and model predictions, is presented in Figure 3.3.

MathML representations are obtained using LaTeXML[3], a widely adopted tool for converting LaTeX documents into XML-based representations. LaTeXML has been extensively used by the scientific community, notably by arXiv, to transform LaTeX sources into HTML formats for web-based visualization of scientific articles[4].

[3] https://math.nist.gov/~BMiller/LaTeXML/, accessed 2026-07-30.
[4] https://info.arxiv.org/about/accessible_HTML.html, accessed 2026-08-17.

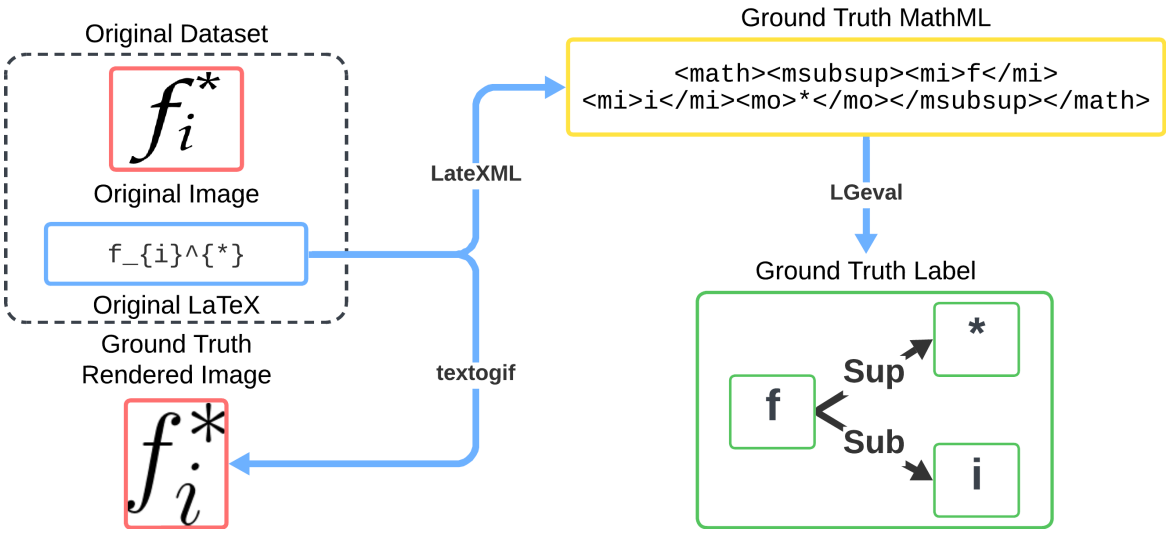


Figure 3.2: Our conversion pipeline from the original dataset sample to MathML, Label Graph, and rendered image, illustrated for the expression f_i^*.

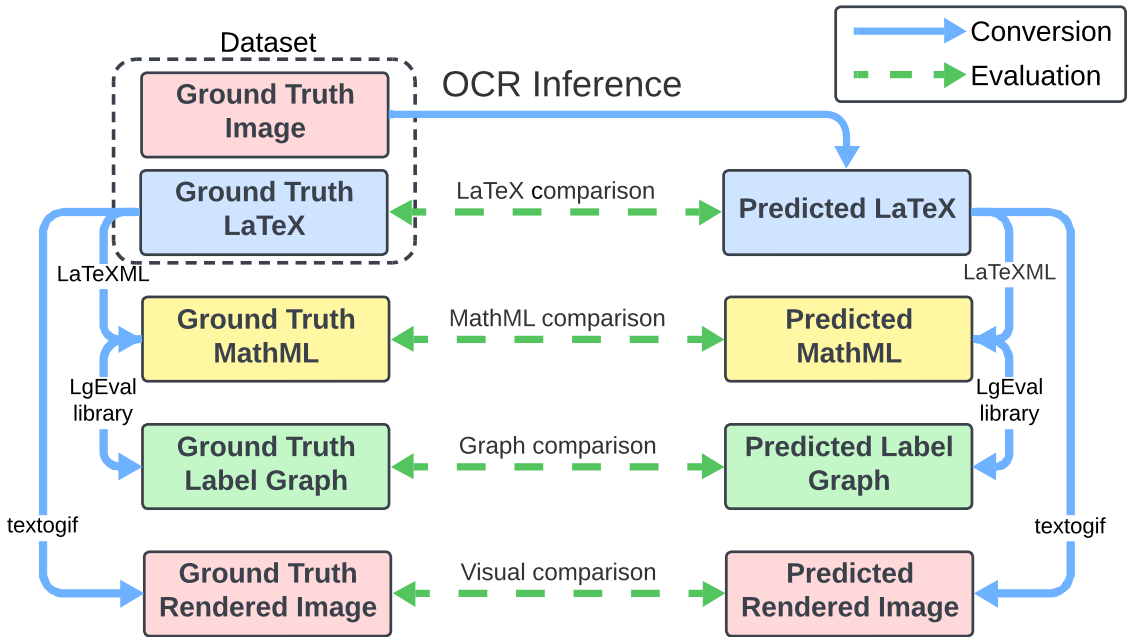


Figure 3.3: Our conversion pipeline, converting the ground-truth dataset and predictions to obtain the four evaluation modalities: LaTeX, MathML, Label Graph, and rendered image.

The conversion from LaTeX to MathML reduces some syntactic ambiguities inherent to LaTeX. For example, the two LaTeX expressions `x^a_b` and `x_b^a` are syntactically different but represent the same mathematical structure. In MathML, both are represented using the same structure with the unique element `msubsup`, as `<msubsup><mi>x</mi><mi>b</mi><mi>a</mi></msubsup>`, removing this ordering ambiguity. Two expressions considered different under LaTeX Exact Match may become identical after conversion to MathML. Additionally, we apply normalization procedures to standardize representations, including Unicode character normalization, by replacing character representations with their HTML form (e.g., using the HTML entity 'β' instead of the Unicode character β).

Label Graph representations are generated using the LgEval library[5]. It converts MathML expressions into graphs describing both the symbols and their positional relationships. Label Graphs provide a structural representation of mathematical expressions, where ambiguities related to LaTeX syntax are further reduced. By explicitly representing spatial and semantic relationships between mathematical symbols, this modality enables a direct analysis of the structural correctness of predicted expressions.

Normalized images are generated by rendering LaTeX expressions using the formula2image tool introduced by Anssi "Miffyli" Kanervisto[6]. The generated images are rendered at 200 DPI to provide a common evaluation resolution. Image-based evaluation is particularly relevant because purely textual metrics are inherently limited by polymorphic ambiguities, where multiple LaTeX expressions produce identical mathematical renderings. Since both the ground-truth and predicted expressions are available during evaluation, they can be rendered using exactly the same rendering process and parameters, allowing for a fair visual comparison that is independent of the original image source characteristics. This approach relies on access to the ground-truth LaTeX expression. In a production scenario, only the predicted LaTeX expression is available, making it impossible to generate a normalized rendering of the ground-truth. Instead, the prediction must be compared with the original reference image, which may contain arbitrary fonts, resolutions, noise, scanning artifacts, or other acquisition-related variations. This limitation is discussed in Chapter 5.

If one of the conversion processes fails, the corresponding modality is replaced by an empty string and receives the lowest possible evaluation score.

Using these four modalities, the proposed evaluation framework provides complementary views of mathematical expression correctness: visual similarity through rendered images, structural similarity through MathML and Label Graph representations, and syntactic similarity through LaTeX representations.

[5] https://gitlab.com/dprl/lgeval, accessed 2026-07-30.

[6] https://github.com/Miffyli/im2latex-dataset/blob/master/src/formula2image.py, accessed 2026-08-22.

Metric Selection

To evaluate model predictions across the four modalities, we first define four reference metrics based on the Exact Match ratio between predictions and ground-truth for each representation: LaTeX Exact Match, MathML Exact Match, Label Graph Exact Match, and Visual Exact Match. These strict metrics represent the percentage of predictions that are considered perfectly identical to the reference representation. Although Exact Match is a very restrictive metric, it provides a clear binary target indicating whether a prediction is fully correct under a given modality. These values are used as reference tasks for correlation analyses to identify which softer metrics best reflect complete correctness.

For LaTeX and MathML representations, we use token-wise Levenshtein distance and BLEU score, two commonly used metrics in MER evaluation.

For the image modality, we compute the Image Levenshtein distance [26] and a Pixel Difference Ratio, measuring the pixel-level difference between normalized rendered images. The Image Levenshtein distance represents each vertical pixel slice of the images as a character and computes the Levenshtein distance between the resulting sequences. The Pixel Difference Ratio is computed as the number of differing pixels between the two images divided by the total number of pixels.

For Label Graph representations, we adapt the $\Delta E(\%)$ metric provided by LgEval. The original metric includes a ΔS component designed to evaluate stroke matching in handwritten MER, which is not relevant for our printed MER evaluation scenario. Therefore, we exclude this component and define the ΔE_{symlg} metric:

$$\Delta E_{\text{symlg}} = \frac{1}{2}\left(\frac{\Delta C}{n} + \sqrt{\frac{\Delta L}{n(n-1)}}\right),$$

where ΔC measures classification errors, i.e. the number of misrecognized symbols, and ΔL measures structural relationship errors between the symbols. This metric provides a compact measurement of both symbol and structural differences. Although LgEval provides several graph-based metrics, we focus on this single metric to simplify the analysis of the Label Graph modality.

The proposed framework can easily integrate additional metrics. However, in this large-scale evaluation involving multiple datasets and models, the computational cost of computing a large number of metrics can become significant. That is why we limit the evaluation to this set of metrics covering multiple aspects of mathematical expression similarity.

To identify the most relevant metrics for each Exact Match target, we compute the point-biserial correlation between each continuous metric and the corresponding binary Exact Match metric. It provides a measure of how well each metric reflects the exact recognition success.

For consistency, all distance-based metrics are normalized between 0 and 1, and converted into similarity scores using $1 - \textit{normalized distance value}$ before correlation analysis. This ensures that all metrics follow the same direction, where higher values always indicate better agreement between prediction and reference.

Table 3.3: Summary of selected evaluation metrics across the four modalities.

Modality	**Metrics**
LaTeX	Exact Match, Levenshtein, BLEU
MathML	Exact Match, Levenshtein, BLEU
Image	Exact Match, Image Levenshtein, Pixel Difference Ratio
Label Graph	Exact Match, $\Delta E(\%)$, ΔE_{symlg}

3.4 Results

In this section, we present the results obtained with our evaluation pipeline. We must first confirm that our results align with those reported by the authors. We examine the effect of mathematical expression normalization on LaTeX and MathML recognition evaluation, before exploring different metrics based on their correlation with target metrics. Finally, we measure the influence of model architectures and training datasets on generalization capabilities. A more in-depth discussion of the observed trends and their implications is developed in section 3.5.

3.4.1 Validation of Models' Performance

Before conducting model evaluations and comparisons, we first verify that our benchmark reproduces, as closely as possible, the results reported by the original authors. This validation step provides evidence that the evaluation pipeline produces results consistent with those reported by the model authors, despite differences in datasets and evaluation conditions. It provides a basis for using the pipeline in the following comparative experiments.

Each model was evaluated both with (**Rescale**) and without (**Base**) our adaptive input rescaling strategy to assess the influence of input DPI on MER accuracy.

For each dataset, we computed the average edit score between the ground-truth and the predicted expressions across the four modalities: Token-wise Levenshtein Score for the LaTeX and MathML modalities, Image Levenshtein Score for the image modality, and ΔE_{symlg} for the Symbol Label Graph modality. To compute an overall score, we used the macro-average, which gives equal weight to each dataset regardless of its size. This prevents large datasets such as IBEM (with more than 44,000 expressions), from

dominating the final score over much smaller datasets such as realFormula (with 121 expressions).

The final scores, reported in Table 3.4, highlight the impact of adaptive rescaling on model performance.

Table 3.4: Impact of image rescaling on OCR performance across metrics and datasets. Best values per model in **bold**. Rescaling enhances prediction quality for MathNet and KYS, while it degrades performance for the other models.

Average score for each modality (%)	MathNet		KYS		TexTeller		pix2tex		Texify	
	Base	Rescale	Base	Rescale	Base	Rescale	Base	Rescale	Base	Rescale
Image Levenshtein	32.7	**72.6**	18.9	**53.2**	**52.6**	39.6	**66.1**	56.4	**75.6**	60.2
LaTeX Levenshtein	45.0	**87.0**	42.4	**67.5**	**71.8**	50.2	**75.3**	70.7	**81.9**	71.2
MathML Levenshtein	52.2	**93.5**	48.6	**73.3**	**82.3**	59.6	**86.3**	82.8	**92.1**	79.6
Label Graph ΔE_{symlg}	54.5	**90.2**	55.3	**78.2**	**79.9**	61.2	**83.9**	78.0	**89.0**	80.2
Average Improvement	**+90.0%**		**+83.2%**		**-26.5%**		**-8.0%**		**-14.2%**	

Adaptive rescaling significantly improved MathNet and KYS predictions, enhancing overall edit scores by 90% and 83.2% respectively. However, it degraded performance for TexTeller, pix2tex, and Texify. Since MathNet and KYS lack internal rescalers, they require external resizing when the image DPI does not match their training resolution, as their image preprocessing is designed to handle only this specific resolution. On the other hand, Texify, TexTeller, and pix2tex include internal rescalers that can handle any input resolution. This makes prior rescaling counterproductive, as performing two consecutive up or downsampling inevitably leads to a greater loss of visual information.

In the remaining evaluations, MathNet and KYS will be used with our adaptive rescaling, as it is necessary for them to achieve consistent performance across heterogeneous datasets.

With the models performing reliably, we compared our benchmark results to those reported by the authors. Since each model was originally evaluated on different test sets that are not always available, we used the im2latex100k-parsed dataset for a general comparison. Table 3.5 shows the LaTeX BLEU and Levenshtein scores for each model, comparing the authors' reported performance when available with our benchmark results.

Table 3.5: Comparison of Levenshtein and BLEU scores between authors' results on their reference test set and ours on the im2latex100k-parsed dataset.

Model	Reference Test Set	Levenshtein Score (%)		BLEU (%)	
		Reference	Ours	Reference	Ours
KYS	i2l100k	83.00	85.86	–	80.96
pix2tex	i2l100k, arXiv, Wiki	90.00	88.62	88.00	83.05
Texify	i2l100k, arXiv	93.50	90.22	84.20	85.07
TexTeller	TexTeller test set	–	72.93	–	59.02
MathNet	i2l100k	88.60	84.35	86.00	74.76

Most of the scores are within 5% of the reported values, supporting the validity of our evaluation tool. The exception is MathNet's BLEU score, which is 11 points lower, likely due to variations in tokenization or metric computation. Undisclosed dataset preprocessing and image handling may explain minor discrepancies, but overall rankings remain consistent, providing evidence that our evaluation pipeline produces results compatible with previously reported performance.

3.4.2 Impact of Normalization: LaTeX Parser vs. MathML Conversion

Due to the polymorphic ambiguity of LaTeX representations, string-based metrics such as Levenshtein score can provide misleading evaluations, as different LaTeX expressions may produce visually identical mathematical renderings. This section investigates how normalization strategies affect model evaluation and whether they can reduce this evaluation noise.

Specifically, we compare two normalization approaches. The first one relies on the LaTeX parser introduced by Deng et al. [22], which transforms raw LaTeX expressions into a normalized representation by reducing syntactic variations. The second approach consists of converting LaTeX expressions into MathML using LaTeXML. As described in Section 3.3, MathML conversion aims to provide a more standardized representation of mathematical expressions and also constitutes an interesting modality for designing evaluation metrics. Therefore, this experiment investigates whether MathML conversion can provide an equivalent normalization effect to the LaTeX parser, while offering an additional structural representation of the expression.

To perform this analysis, predictions from the five evaluated models were compared against three reference representations derived from the im2latex100k dataset: im2latex100k-raw, corresponding to the original LaTeX annotations without any normalization; im2latex100k-parsed, obtained by applying the LaTeX parser from Deng et al. to the raw annotations; and MathML, obtained by converting the raw LaTeX annotations into MathML using LaTeXML.

Figure 3.4 illustrates the transformation process. Starting from the original LaTeX representation, two independent normalization paths are considered: LaTeX parsing and MathML conversion. The objective is to determine whether both approaches lead to similar evaluation results, which would indicate that MathML conversion can effectively replace the dedicated LaTeX parsing step for reducing polymorphic ambiguity.

Table 3.6 presents the Levenshtein scores obtained with these different reference representations. The comparison allows us to measure the impact of each normalization strategy and determine whether MathML provides a comparable reduction of evaluation variability to the LaTeX parser.

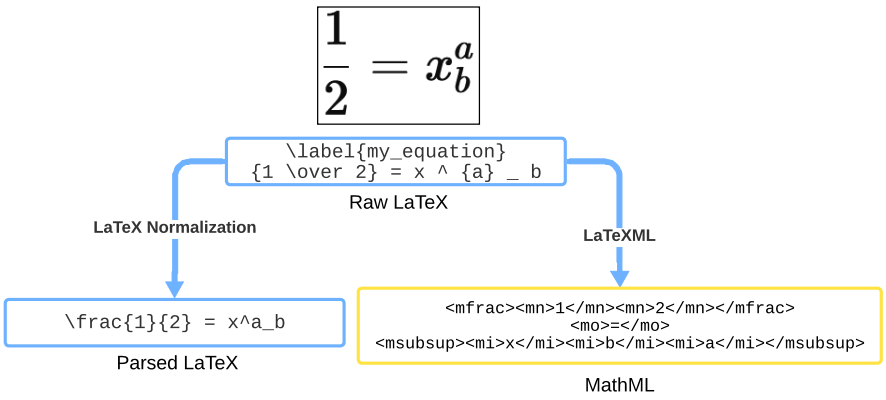


Figure 3.4: Comparison of normalization strategies: LaTeX parsing (Deng et al.) and MathML conversion (LaTeXML).

Table 3.6: Levenshtein scores (%) between model predictions and reference data of im2latex100k in different formats. Scores on parsed LaTeX and converted MathML are both consistently higher than on raw LaTeX.

Model	Raw LaTeX	Parsed LaTeX Δ	MathML Δ
KYS	61.89	85.86 **(+23.97)**	85.05 (+23.16)
MathNet	63.08	84.35 (+21.27)	89.95 **(+26.87)**
pix2tex	63.36	88.62 (+25.26)	91.09 **(+27.73)**
Texify	67.03	90.22 (+23.19)	96.06 **(+29.03)**
TexTeller	54.05	72.93 (+18.88)	80.50 **(+26.45)**

Normalization significantly improves the Levenshtein scores. Both parsed LaTeX and MathML conversion yield higher scores than raw LaTeX, with MathML generally matching or surpassing Parsed LaTeX in most cases. This suggests that MathML is a reliable modality for model evaluation.

3.4.3 Correlation of Metrics with Target Exact Matches

This section reports the point-biserial correlation of our eight metrics with each target Exact Match. Point-biserial correlation measures the relationship between continuous (evaluation metric) and binary (Exact Match) variables. The results in Figure 3.5 are sorted by maximum correlation per row to highlight the most informative metrics.

For Visual Exact Match, Image Levenshtein (0.78) and Image Pixel Difference (0.53) show the highest correlations. Other metrics correlate more strongly with MathML or Label Graph Exact Matches, with ΔE_{symlg} showing the second strongest correlation of the set. Levenshtein-based metrics show the strongest correlations for LaTeX Exact Match, while LaTeX and MathML BLEU have weak correlations (<0.40). Image Levenshtein and LaTeX Levenshtein show strong correlations across all target metrics.

LaTeX correlations are weaker mostly because of polymorphic ambiguity. The multiple ways of writing the same expression makes Exact Matches harder to obtain.

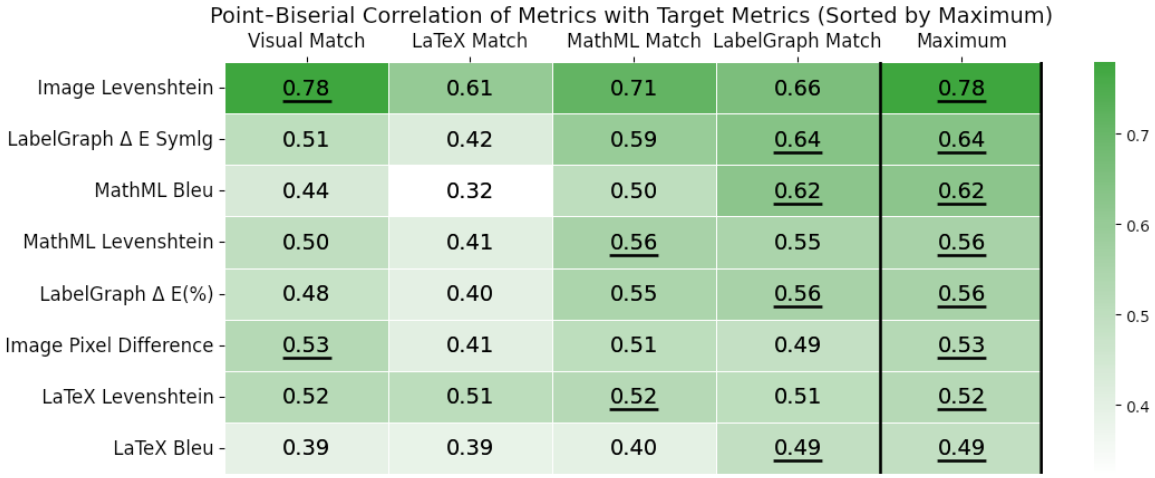


Figure 3.5: Point-biserial correlation coefficients between Exact Matches and a set of metrics, sorted by maximum correlation per row. Higher values indicate the metric aligns with Exact Match for that modality.

MathML, Label Graph, and images are normalized by our sequential conversion process, resulting in stronger correlations. The Image Levenshtein metric outperforms Image Pixel Difference, likely because it captures the spatial organization of the rendered expression better than a pixel-wise difference. Removing the stroke-related component of $\Delta E(\%)$ led to a stronger ΔE_{symlg} metric tailored for printed MER.

3.4.4 Overview of Models' Results on All Datasets

Figure 3.6 displays the individual results of all models across the datasets for the image and LaTeX Levenshtein scores, as these metrics show the strongest correlations with Exact Matches across the four modalities, making them strong candidates for holistic evaluation metrics.

In the LaTeX Levenshtein score heatmap, all models show poorer performance on the im2latex100k-raw dataset, while performing consistently on the other im2latex-derived datasets. This performance drop on im2latex-raw is not observed in the Image Levenshtein heatmap.

Significant differences emerge on the other datasets. On im2latexv2 and realFormula, Texify, MathNet, and to a lesser extent TexTeller, achieve good results, whereas KYS and pix2tex struggle considerably. On ME20k, KYS lags behind in terms of performance. On IBEM, only MathNet excels, with all other models performing notably worse.

Based on the average performance, MathNet achieves the highest LaTeX Levenshtein score, while Texify leads in Image Levenshtein score. pix2tex performs at an intermediate level, whereas TexTeller and KYS show weaker results on the two metrics. Selecting the best-performing model for each dataset yields an average edit score of 89%.

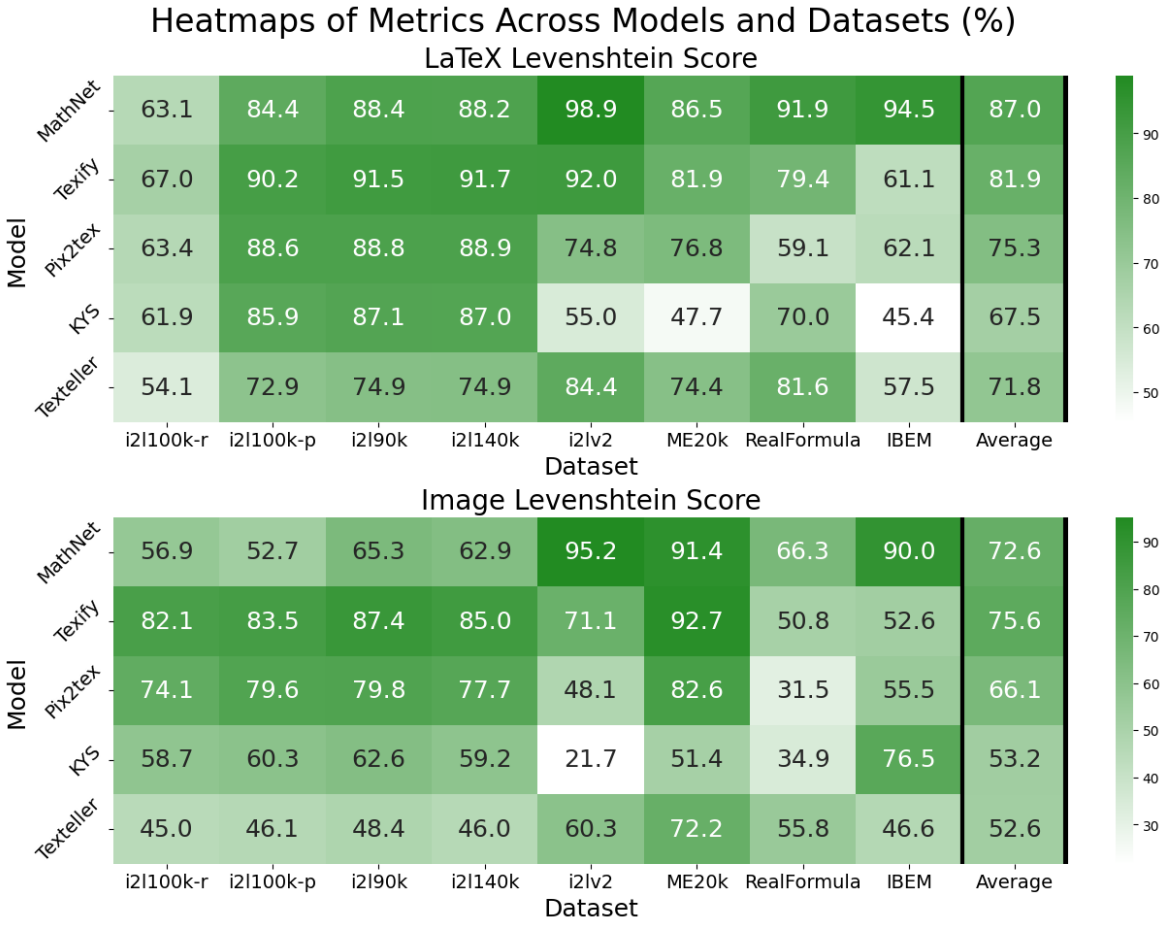


Figure 3.6: Heatmap showing individual results for each dataset–model pair for two metrics. Darker and lighter cells indicate better and poorer performance, respectively. The last column displays the average score per row.

These results highlight large variation in model performance across datasets, indicating that conclusions drawn from a single benchmark may not generalize to other data distributions or rendering conditions.

3.5 Discussion

3.5.1 Image Resolution and Fair Model Evaluation

Understanding and adapting to each model's input requirements is required to achieve optimal performance. In our evaluation, images could be rescaled appropriately because the original DPI was known. In practical applications involving multiple image sources, such as scanned documents or user screenshots, the input resolution is often unavailable. MathNet and KYS both use a fixed input DPI, which would make them harder to use in real-world applications.

Our results on im2latex100k-parsed remain very close to the published ones and preserves the relative ranking of the models. This suggests that im2latex100k-parsed provides a reliable and practical benchmark for estimating the performance of printed MER systems. It is a convenient reference dataset for reproducible comparisons across models. It is however composed of cleanly rendered mathematical expressions extracted from arXiv documents and does not capture the variability encountered in real-world scenarios, including scanning artifacts, compression, noise, diverse fonts, or document degradation. Strong performance on this benchmark should not be interpreted as evidence of robust performance across all practical MER use cases.

We observed a small discrepancy in the BLEU score for MathNet. It may originate from differences in BLEU computation, dataset preprocessing, or LaTeX tokenization. Despite this, our overall findings suggest that most models perform as reported, and these minor differences do not invalidate our analysis.

3.5.2 Preprocessing and Normalization for Fair Evaluation

Levenshtein scores computed on the predictions converted to MathML are generally better than the scores computed on parsed LaTeX ground-truth. This shows that MathML normalization is often more effective at reducing ambiguity than LaTeX parsing. This can be explained by the fact that both the ground-truth and the predictions are converted into MathML, meaning that both representations benefit from the implicit normalization and cleaning of the conversion (see Figure 3.3). MathML is also less syntax-dependent than LaTeX, contributing to better Levenshtein scores. Comparing predictions to raw LaTeX leads to an underestimation of model performance because of polymorphic ambiguity and should be avoided.

We recommend that future authors document the versions of training datasets used as well as any parser or normalization choices, to ensure fair comparisons and prevent discrepancies across evaluations. We recommend LaTeXML for MathML conversions, as it is a long-standing project benefiting from community improvements. Consistent tokenization and preprocessing would also improve the reproducibility of evaluations.

3.5.3 Metrics Alignment with Evaluation Objectives

Metric correlation analysis provides insights into how different metrics align with the Exact Matches for each modality. We summarize our recommendations of metrics for each evaluation objective in Table 3.7. This selection is based on the values of the correlation coefficients between each metric and the Exact Match scores across different modalities. Our correlation analysis shows that commonly used metrics such as LaTeX BLEU and Levenshtein do not correctly track Exact Match, especially when representational ambiguity is present.

Table 3.7: Recommended metrics for different evaluation objectives. This selection is based on our analysis of metric correlations with Exact Match scores across different modalities.

Evaluation Objective	Relevant Metrics
Overall Evaluation	Image Levenshtein, LaTeX Levenshtein
Visual Similarity	Image Levenshtein, Image Pixel Difference
Syntactic Accuracy	LaTeX Levenshtein, Label Graph ΔE_{symlg}
Structural Accuracy	Label Graph ΔE_{symlg}, MathML Levenshtein

The choice of evaluation metrics should be guided by the context for which MER is needed. When the goal is visual similarity (for example in book publishing and other printed materials), small syntactic variations like sub and superscript ordering are irrelevant to the objective. For mathematical search engines, the target is for the model to produce a textual representation that allows for correct retrieval of formulas. Structural correctness is more important than rendering accuracy in this case. In symbolic computation systems such as SymPy or MATLAB, syntactic accuracy is crucial, as any syntax errors could prevent computations from executing properly.

The four modalities analyzed in this work help guide metric selection based on specific task requirements. Correlation analysis also determines which metrics are most relevant for each modality, but considering all available metrics remains valuable for a broad evaluation.

3.5.4 Relationship Between Architecture, Training Data, and Generalization

Training data and model architecture strongly affect generalization capabilities. MathNet's CvT architecture and multi-font training give it strong performance, even on datasets without font variability. In contrast, despite its 7.5M-sample training set, TexTeller performs unreliably, though it still performed well on im2latexv2's multi-font expressions. This suggests dataset size alone is not sufficient for a generalist model.

Dataset characteristics also play a crucial role in evaluation. IBEM and ME20k respectively introduce inline expressions with subtle rendering differences, and expressions that are not from arXiv papers. This greatly affects performance, particularly in IBEM, where short inline expressions cause large fluctuations in normalized Levenshtein scores as a single error can disproportionately impact the metric.

The poor LaTeX Levenshtein score results on the unparsed im2latex100k-raw dataset show the importance of preprocessing. Evaluating on images helps reduce LaTeX polymorphic ambiguity. The visual representation does not rely on specific tokenization or parsing conventions, making it more robust to variations in syntax.

3.5.5 Usage of PiE-MER for other modalities

The principle of this tool could be extended to other recognition tasks by evaluating predictions through multiple modalities, rather than relying only on the modality provided by the dataset.

For table recognition, a dataset containing image–HTML pairs could be evaluated by converting the HTML into alternative representations, such as rendered tables, Markdown, LaTeX, CSV, or structured JSON, and by applying specific metrics to each of them, such as TEDS [161], GriTS [162], and cell-level precision, recall, or F1-score. This would allow the evaluation to measure the quality of the recognized table with respect to its content, layout, and structure.

Similarly, for chemical structure recognition, a dataset of image–SMILES pairs could be evaluated with alternative representations of the same chemical structure, such as InChI, molecular graphs, 3D models, or rendered molecular structures. Predictions could then be compared using metrics such as SMILES Exact Match, molecular validity or Tanimoto similarity [163].

Figure 3.7 illustrates this principle for table and chemical structure recognition. This demonstrates that the proposed methodology is not limited to printed mathematical expressions, but can provide a general framework for multimodal evaluation of structured visual recognition systems.

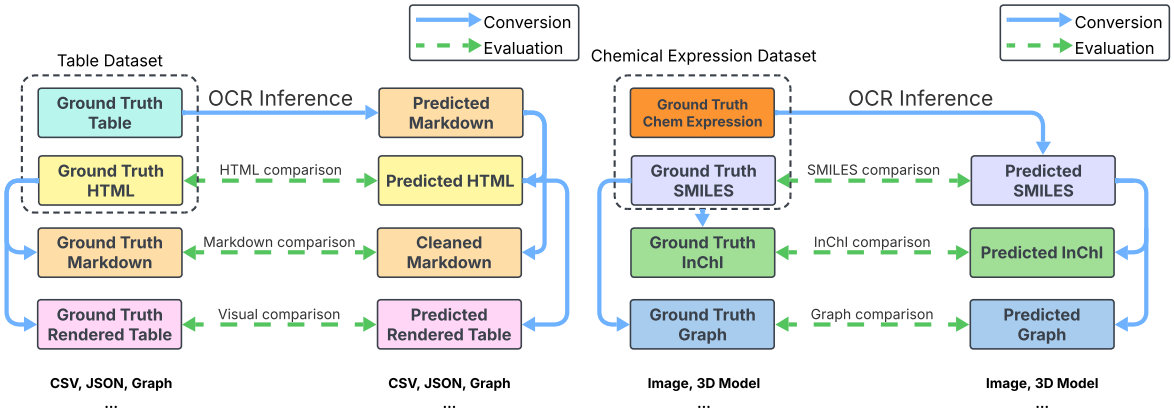


Figure 3.7: Possible adaptation of our framework for the evaluation of table recognition and chemical structure recognition.

3.6 Conclusion

In this chapter, we demonstrated the importance of multimodal evaluation for MER. We introduced a robust evaluation framework using LaTeX, MathML, Label Graphs, and visual representations for a multimodal and task-aligned evaluation of model performance. PiE-MER is available online at https://github.com/fwieckowiak/PiE-MER.

By analyzing metric correlations with modality-wise Exact Matches, we showed the limitations of traditional metrics like BLEU and Levenshtein, and recommended more suitable metrics for specific evaluation goals. Our evaluation of five models across eight datasets shows that preprocessing, normalization, and proper input image resolution improve evaluation consistency. No single model outperforms all others across every modality, reinforcing the importance of aligning training strategies and architectures with the target application and evaluation criteria.

In Chapter 4, we will use the proposed evaluation framework to analyze the performance of our own MER model, which is designed to handle mathematical expressions from patent documents, using a new dataset of mathematical expressions written in MathML. To use the proposed evaluation framework, we added a *MathML mode* to PiE-MER. We avoided performing the conversion from MathML to LaTeX, as we did not find any suitable tool for this conversion. Also, to emulate the normalization process that the LaTeX to MathML conversion would typically involve, we implemented a custom normalization step for MathML markup, especially for handling the specific MathML syntax of the EPO data, considering this "Cleaned MathML" as a new modality. The resulting conversion pipeline is illustrated in Figure 3.8.

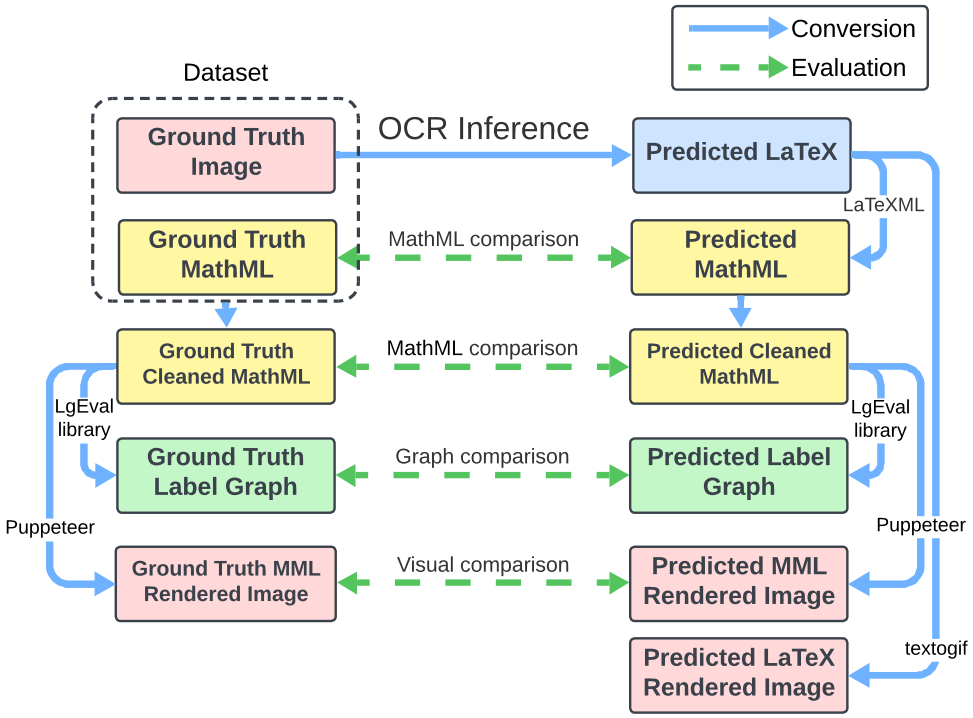


Figure 3.8: Conversion pipeline for the MathML mode of PiE-MER. The original dataset sample is composed of a MathML expression and a raw image. The pipeline generates the image and Label Graph representations, which are used for evaluation.

Chapter 4

PatentME-600k and MML-Net: A Large-Scale Dataset and Model for Patent Mathematical Expression Recognition

4.1 Introduction

The state of the art presented in Chapter 2 highlighted a significant gap in MER for patent applications. Existing models are often trained on mathematical expressions extracted from scientific publications and rendered from LaTeX sources. Some more heterogeneous large-scale datasets have been proposed, but none specifically contains mathematical expressions extracted from patent applications. They can contain heterogeneous fonts, scanning artifacts, compression noise, and other image degradations. During the publication process, these patent expressions must be converted into MathML so that they can be accurately indexed and searched in patent databases. This conversion constitutes one of the challenges addressed by Luminess, the industrial partner of this PhD.

To develop a model to perform this recognition, we first introduce **PatentME-600k**, the first large-scale public dataset of mathematical expressions extracted from European patent applications. It contains more than 600,000 image–MathML pairs obtained through an automated extraction pipeline from patent XML documents. We describe in Section 4.2 the complete extraction process, the normalization steps applied to the original MathML annotations, and present a statistical analysis of the resulting dataset. **PatentME-600k** is available on Zenodo[1], and the code used to generate it is available at `github.com/fwieckowiak/PatentME`.

We then present **MML-Net** in Section 4.3, a new encoder–decoder OCR model trained on PatentME-600k. Unlike existing MER systems that generate LaTeX, MML-Net directly generates MathML, making it directly compatible with the patent publication workflows while avoiding an additional conversion step.

Finally, MML-Net is evaluated using the PiE-MER evaluation framework, presented in the Chapter 3, in Section 4.4. Since MML-Net predicts MathML markup, PiE-MER is augmented with a dedicated MathML evaluation mode that includes a normalization stage adapted to the EPO annotations and MathML-specific conversions. The proposed model is compared with several state-of-the-art MER systems on both PatentME-600k and public benchmark datasets to measure its performance and generalization capabilities.

[1] `https://zenodo.org/records/20446086`, accessed 2026-08-03.

4.2 The PatentME-600k Dataset

4.2.1 Source of the Patent Collection

The patent collection used in this chapter was obtained from the *EP Full-text Data* distributed by the EPO. This collection contains all EP-A and EP-B publications released by the EPO from the 1970s to the present. Before creating the dataset, the EPO was contacted to confirm that the extracted mathematical expression images and MathML annotations could be redistributed for research purposes. The dataset documentation includes the required EPO attribution statement.

The data are distributed as weekly ZIP archives, each containing all patent publications of the week. We named the archives according to the publication year and week number. For example, *EP Full-text Data 2026/031* contains all European patent publications released during the 31st week of 2026. Each ZIP archive is organized into folders corresponding to the publication kind codes. Documents of type *A* correspond to patent applications (e.g., A1, A2, etc.), while documents of type *B* correspond to granted patents (e.g., B1, B2, etc.). Additional suffixes such as *NW*, *W1*, and *W2* indicate corrected versions of previously published documents.[2]

Within each publication–type directory, individual patent documents are stored in separate archives. Each archive contains the XML representation of the publication, its PDF version, and all extracted assets, including the figures, chemical expressions, and segmented images of mathematical expressions originating from the original patent application manuscript. Figure 4.1 illustrates the directory structure of an EPO weekly ZIP archive, while Figure 4.2 presents the content of a single patent folder.

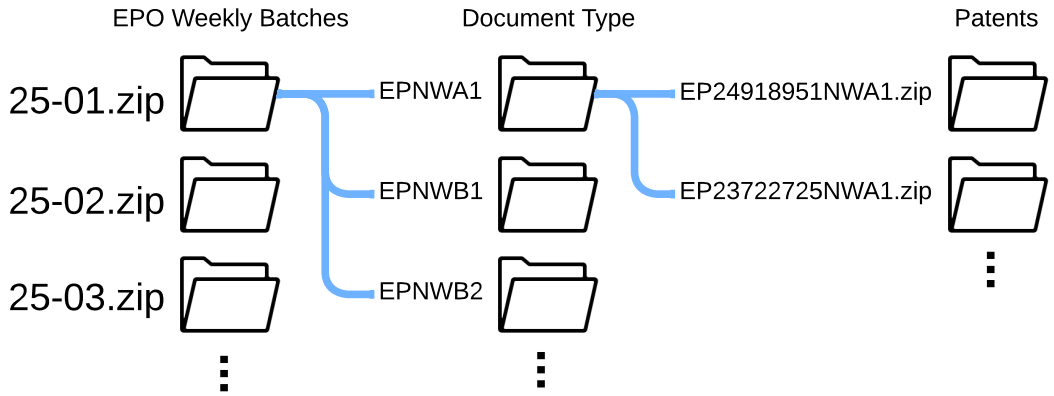


Figure 4.1: Directory structure of an EPO weekly *EP Full-text Data* ZIP archive.

This source was selected because the EPO provides an accessible way to retrieve large-scale patent collections. Weekly archives are publicly available through stable download

[2]See https://www.epo.org/en/searching-for-patents/helpful-resources/first-time-here/definitions for more details (accessed 2026-08-04).

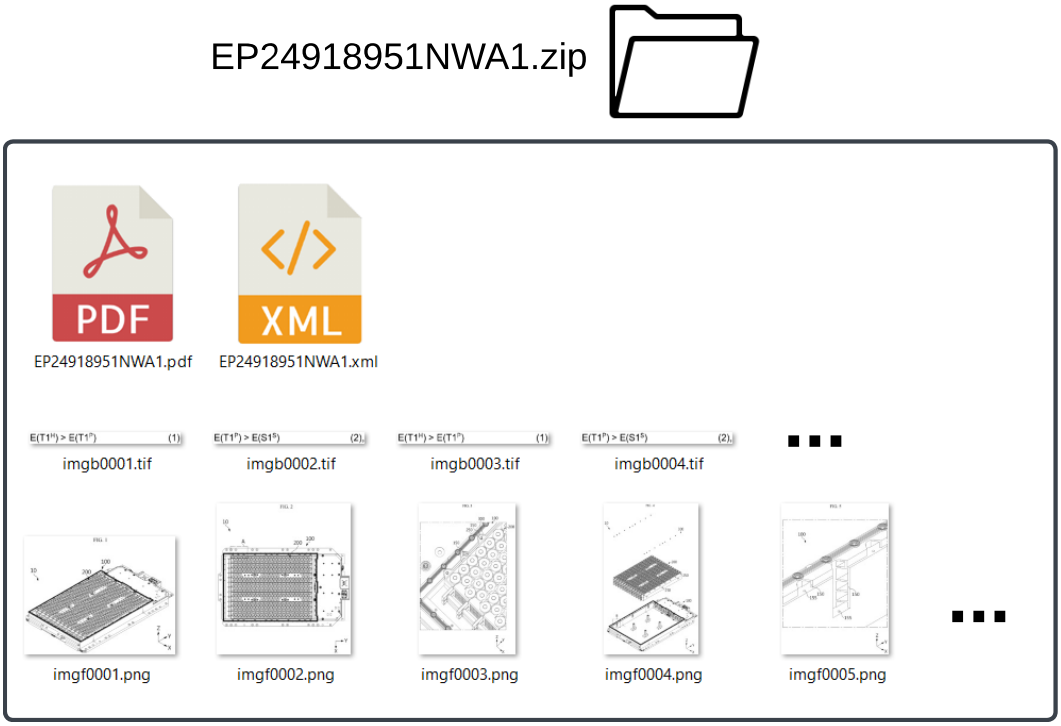


Figure 4.2: Example of the contents of a single patent archive within an EPO weekly archive. The example is simplified for illustration purposes and does not correspond to an actual patent publication.

links, allowing the automatic acquisition of a large number of patent documents.

To train our MER model, we need to create image–MathML pairs for mathematical expressions. While the segmented mathematical expression images are directly available inside each patent folder, the corresponding MathML annotations are embedded in the XML representation of the publication. A dedicated extraction pipeline was developed to recover these annotations and associate them with their corresponding images.

The pipeline processes each weekly ZIP archive and extracts all individual patent applications. For each of them, the XML file is parsed and all mathematical expressions are identified through the `<maths>` elements. Each mathematical expression contains a `<math>` element storing the MathML annotation and an `<img>` element containing the reference to the associated TIF image. The image–MathML matching is performed using the image filename stored in the XML `file` attribute.

Figure 4.3 illustrates an example of this matching process between the patent PDF document, the XML annotation, and the extracted TIF mathematical expression image.

For each extracted expression, a selection of metadata is saved alongside the image and MathML annotation, as summarized in Table 4.1.

The complete extraction pipeline is applied to all EPO weekly archives from batch 25-01 to batch 26-11, corresponding to 63 weekly archives (52 weeks of 2025 and 11 weeks of 2026). This period was selected as it represented the most recent available

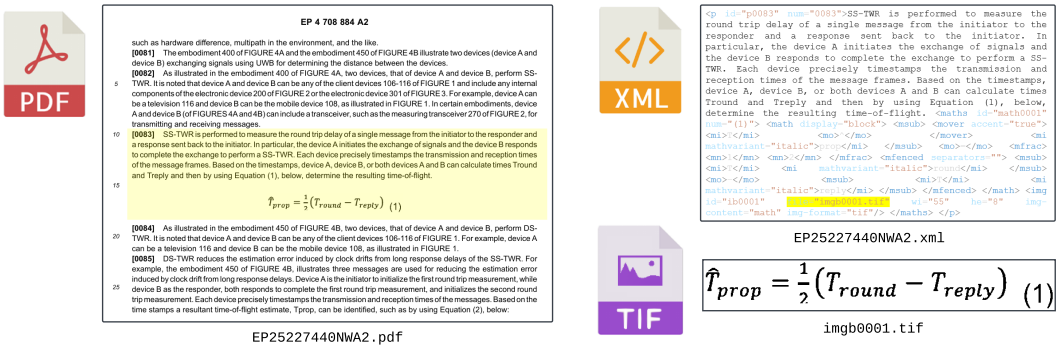


Figure 4.3: Example of a mathematical expression extracted from a patent PDF, its associated XML representation, and the corresponding image matched using the `file` attribute of the XML annotation.

Field	Description
`image_path`	Path to the extracted mathematical expression image
`patent_number`	Identifier of the source patent document
`math_id`	Identifier of the mathematical expression
`img_file`	Original image filename
`display_type`	Display mode of the mathematical expression, inline or block
`img_width`, `img_height`	Image dimensions provided by the XML metadata
`real_img_width`, `real_img_height`	Actual dimensions of the image (slightly bigger)
`img_id`	Unique identifier combining patent number and image name
`mathml`	Extracted MathML representation of the expression

Table 4.1: Metadata stored for each extracted mathematical expression.

collection at the time of dataset creation while already providing a sufficiently large number of documents.

The final collection contains 391,824 patent applications. Among them, only 54,022 patents contain at least one mathematical expression (13.79%), while 337,802 patents do not contain any (86.21%), which justifies the need for large-scale automatic extraction to generate enough mathematical expressions. Since EPO patents can be published in English, French, or German, extracted expressions may also contain textual elements from these languages. Therefore, the dataset includes mathematical expressions with both symbolic content and occasional textual components (See Figure 4.3).

The complete dataset is divided into training, validation, and test subsets using an 80%, 10%, 10% split. After extraction, all MathML annotations are processed by the normalization pipeline described in the following section to obtain a consistent representation for training and evaluating MER models.

4.2.2 MathML Normalization and Cleaning

The raw MathML annotations provided in EPO patent documents cannot be directly used for model training and evaluation due to the presence of deprecated MathML tags and variations in the representation of some equivalent mathematical structures. That is

why we designed a dedicated MathML cleaning and normalization pipeline. Its objective is to convert the heterogeneous MathML markups into a consistent representation while preserving their semantic and visual properties.

First, the MathML structure is standardized by replacing deprecated or problematic elements with more robust alternatives. In particular, all `<mstyle>` elements are converted into `<mrow>` elements. When an attribute such as `mathvariant = ...` is defined at the `<mstyle>` level, it is propagated to descendant `<mi>` elements that do not already specify a variant, ensuring that character styling information is preserved after the conversion.

Unnecessary attributes and redundant structures are also removed. The `display` attribute of the root `<math>` element is discarded and saved as a separate metadata. Redundant nested `<mrow>` are also removed.

Deprecated `<mfenced>` elements are converted into equivalent `<mrow>` structures. This transformation is required because `<mfenced>` is deprecated in current MathML specifications and is not properly rendered by all modern software. The opening and closing delimiters are converted into `<mo>` operators, and separators are inserted between child elements. An example for this conversion is given below :

Before

```
1 <mfenced open="(" close=")"
      separators=",">
2     <mi>a</mi>
3     <mi>b</mi>
4 </mfenced>
```

After

```
1 <mrow>
2     <mo>(</mo>
3     <mi>a</mi>
4     <mo>,</mo>
5     <mi>b</mi>
6     <mo>)</mo>
7 </mrow>
```

Spacing information is also normalized. Since `<mspace>` elements may contain variable spacing values depending on the original rendering process, all spacing elements are replaced by explicit non-breaking and standardized spaces represented as `<mtext> </mtext>`. Similarly, empty `<none>` elements used in some MathML are replaced with explicit spacing tokens to avoid information loss during subsequent normalization operations.

Mathematical Unicode characters are normalized to avoid inconsistencies between Unicode-based and MathML-based representations. Characters containing implicit typographic information, such as double-struck, script, italic, bold, or fraktur variants, are decomposed into their base character and an explicit `mathvariant` attribute. For instance, the Unicode character $\mathbb{R}$ is converted to `<mi mathvariant="double-struck">R</mi>`. This normalization ensures that equivalent mathematical symbols share the same representation regardless of their original encoding.

Finally, adjacent identifier elements are merged whenever they represent a continuous identifier. During this operation, the original mathematical style is preserved. For example, two consecutive italic identifiers, `<mi>A</mi><mi>B</mi>`, are normalized into `<mi mathvariant="italic">AB</mi>`. This normalization reduces unnecessary structural variations while maintaining the exact same visual interpretation of the mathematical expression.

Overall, this preprocessing step produces a homogeneous MathML representation from heterogeneous EPO annotations. It reduces formatting-related variability and provides a consistent input representation for MER model training and evaluation.

4.2.3 MathML Tokenization and Vocabulary Construction

After the MathML cleaning and normalization steps, the resulting expressions are converted into token sequences for training our sequence-to-sequence models. We designed the tokenizer to provide a compact representation. Each opening and closing MathML tag is converted into a dedicated token, including the associated attributes when present. For example, the element `<mi mathvariant="italic">` is considered as a single token. The textual contents and words contained inside the MathML elements are separated into individual character and embedded as unique tokens. This hybrid tokenization strategy preserves the structural information of MathML while limiting the vocabulary size. Since spacing elements are normalized during the previous cleaning stage as `<mtext> </mtext>`, these elements are represented by a dedicated single special token `<mspace/>` to simplify their generation. Finally, four additional special tokens are added to the vocabulary: `<PAD>` is used for sequence padding, `<UNK>` represents unknown tokens, and `<SOS>` and `<EOS>` respectively indicate the beginning and the end of a sequence.

Below is an example of the tokenization of a MathML markup, resulting in the token list `[2,61,128,145,134,134,133,34,3]` :

Before

```
<mi mathvariant="italic">
    speed
</mi>
```

After

```
<SOS>
<mi mathvariant="italic">
    s p e e d
</mi>
<EOS>
```

To avoid data leakage, only MathML expressions from the training subset are used to build the vocabulary. During vocabulary construction, token frequencies are computed, and tokens occurring fewer than a predefined minimum number of times are removed. This filtering step eliminates extremely rare tokens to reduce the decoder complexity. We selected a minimum frequency threshold of 9, meaning that a given token must

occur at least nine times in the training set to be included in the vocabulary. This reduces the vocabulary from 438 to 352 tokens, corresponding to a 19.6% reduction in vocabulary size. However, these 86 removed token types account for only 280 token occurrences out of 45,419,180 of the whole dataset, corresponding to a negligible 0.00062% of all token occurrences. The removed tokens include "⊢", "↔" and `<mtable groupalign="decimalpoint">`. The complete list is available in Figure A.1 of the appendix. These removed tokens are treated as `<UNK>` tokens.

The final vocabulary is stored as an ordered list, and two mappings are generated: a token-to-index mapping used during model training and an inverse index-to-token mapping used for sequence decoding. Each normalized MathML expression is converted into a sequence of tokens by replacing every token with its corresponding vocabulary index. The resulting tokenized representations are stored alongside the original MathML expressions and will be used for model training and evaluation.

4.2.4 Statistical Analysis

A statistical analysis was conducted to characterize the size, visual properties, and complexity of the expressions of PatentME-600k. The main statistics are summarized below, while more detailed results are available in Appendix A.1.

PatentME-600k contains 669,582 image–MathML pairs extracted from 51,952 patents. The dataset is divided into training, validation, and test sets with an 80%, 10%, 10% split. The split is performed at the patent level, ensuring that all expressions from a given patent are assigned to the same set. It contains both block and inline mathematical expressions, with block expressions representing 66.1% of the dataset. The resulting dataset statistics are summarized in Table 4.2.

Table 4.2: Overview of the PatentME-600k dataset.

Statistic	**Value**
Number of expressions	669,582
Number of patents	51,952
Training samples	535,665 (80.0%)
Validation samples	66,958 (10.0%)
Test samples	66,959 (10.0%)
Block expressions	442,716 (66.1%)
Inline expressions	226,866 (33.9%)
Vocabulary size	352

MathML complexity and visual properties

The complexity of a mathematical expression can be described from both its visual representation and its MathML representation. For the former, we report the image

width, image height, and image aspect ratio. For the latter, we can measure the tokenized MathML length. Since the MathML expressions were tokenized so that complex tags such as `<mi mathvariant="bold">` are represented as a single token, we avoid measuring artificially long sequence length due to the MathML markup. These distributions are shown in Figure 4.4 and Table 4.3.

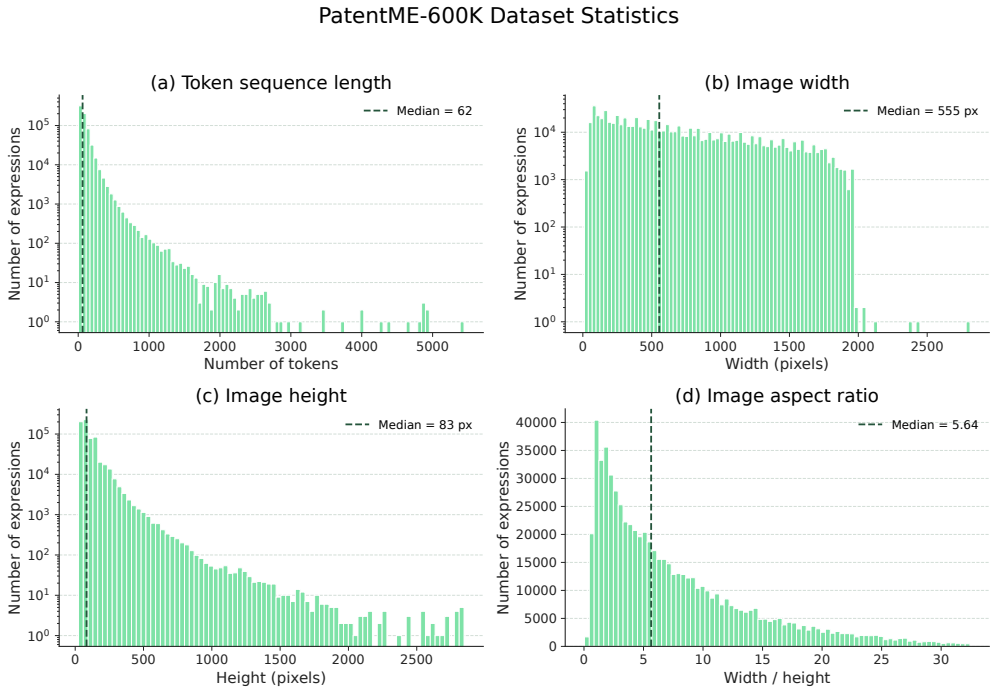


Figure 4.4: Distributions of token sequence length, image width, image height, and image aspect ratio in PatentME-600k.

Table 4.3: Main MathML complexity statistics of PatentME-600k.

Measure	**Mean**	**Median**	**P95**	**Max**
Token sequence length	86.80	62	230	5,443
Image width (px)	677.8	555	1,642	2,811
Image height (px)	108.1	83	248	2,846
Aspect ratio	7.77	5.64	22.22	52.97

The dataset covers a broad range of expression sizes. They have a median token count of 62 tokens and a 95th percentile of 230 tokens, while only 103 expressions (0.02%) contain more than 2,000 tokens. The maximum observed sequence length is 5 443 tokens. The images also show large variations in size, with a median width of 555 pixels, a median height of 83 pixels, and a median aspect ratio (width over height) of 5.64. This large aspect ratio is consistent with the fact that many mathematical expressions are written as horizontal, single-line expressions. A clear cutoff can be observed at approximately 2,000 pixels in image width. This can be explained by the physical dimensions of the source pages. For reference, an A4 page at 300 DPI is

approximately 2,480 pixels wide. This explains why almost all expressions remain below this width (minus 480 pixels for the margins), with only a few exceptions.

MathML Vocabulary and Token Distribution

The train split of the dataset contains 352 distinct tokens. The most frequent tokens are mainly MathML structural tags and common mathematical operators and symbols. Table 4.4 reports the most frequent tokens, while Figure 4.5 shows their rank-frequency distribution.

Table 4.4: The most frequent tokens in the train split of PatentME-600k.

Token	Frequency	Token	Frequency	Token	Frequency
`</math>`	535,665	`<mrow>`	489,678	`<mn>`	373,771
`<math>`	535,665	`</mrow>`	489,678	`</mn>`	373,713
`</mi>`	529,301	`<mi>`	476,557	`=`	328,110
`<mo>`	495,945	`<msub>`	276,144	`</msub>`	276,144
`</mo>`	495,945	`)`	260,020	`(`	259,877

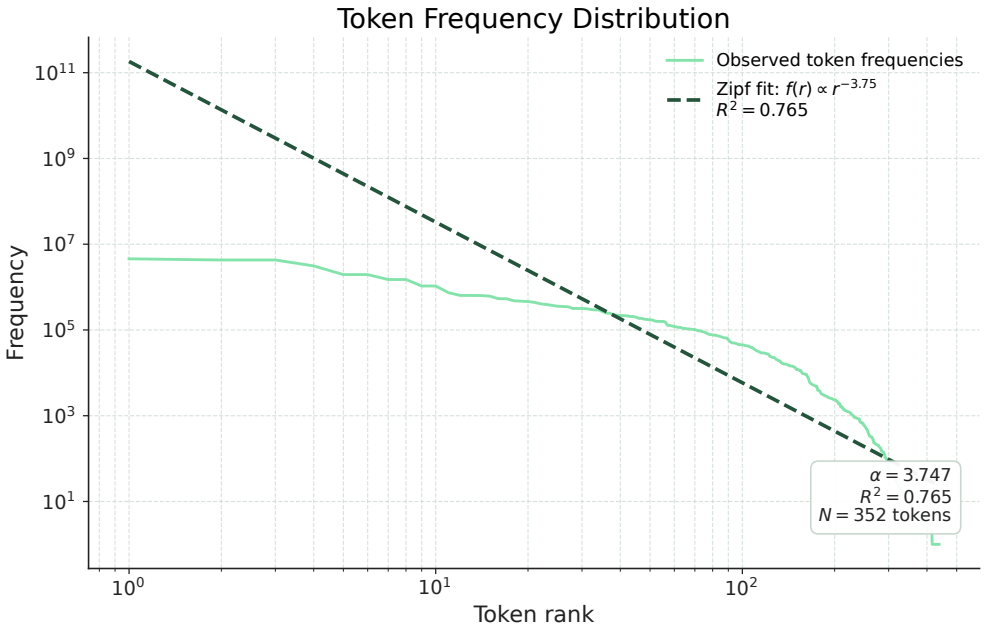


Figure 4.5: Token-frequency distribution in PatentME-600k on logarithmic axes. The dashed line shows the fitted power-law trend.

The frequency distribution has a distinctive shape caused by the syntax of MathML. Unlike natural-language words, many MathML tokens are constrained by markup rules. In particular, opening and closing tags are used in pairs, so tokens such as `<mo>` and `</mo>`, or `<mrow>` and `</mrow>`, occur with very similar frequencies. However, the tokenizer treats opening `mi` tags with attributes, such as `<mi mathvariant="italic">`,

as distinct tokens. This explains the differences in frequency between opening and closing `mi` tags.

In natural language, word distributions typically follow Zipf's law, with frequency decreasing according to a power law and appearing approximately as a straight line on a log-log plot. In contrast, the PatentME-600k token distribution has a curved trend, with a small set of highly frequent tokens followed by a long tail. A power-law fit gives an exponent of approximately -3.75 and an R^2 of only 0.765, indicating a clear deviation from Zipf's law. This difference reflects the fact that MathML tokens do not behave like natural-language words: they are strongly constrained by XML syntax, with opening and closing tags, as well as nested structures, generating characteristic frequency patterns[3].

PatentME-600k provides a large number of training examples with broad variability in visual and structural complexity, which is valuable for training a robust encoder–decoder recognition model. The complete statistical analysis, including additional percentiles, per-split statistics, and structural element frequencies is provided in Appendix A.1.

4.3 MML-Net Model

This section presents MML-Net, a MER model designed to recognize mathematical expressions from patent images and directly generate their MathML representation.

Direct MathML generation reduces the number of transformations required between recognition and subsequent processing. To the best of our knowledge, very few MER systems generate MathML directly, as most existing approaches target LaTeX and would require an additional LaTeX-to-MathML conversion step for patent processing. MML-Net instead learns directly from MathML extracted from patents, producing the representation required by the industrial processing chain and matching the requirements and style of the EPO.

Rather than designing a highly complex model from scratch, we select four architectures from the recent MER literature, train and evaluate them on PatentME-600k and select the best-performing one. It is then further evaluated against existing MER systems on multiple datasets using the PiE-MER evaluation framework presented in Chapter 3.

[3]We highly recommend the video essay by Vsauce, "The Zipf Mystery", 2015. Available at https://youtu.be/fCn8zs912OE, accessed 2026-08-24.

4.3.1 Design Choices

Our objective is not to reach the extremely high accuracy of the existing manual processing chain, which must reach 99.99% accuracy, i.e. Exact Match ratio, after human verification. Such a level of performance is highly unlikely to be reached by any MER model. Instead, the objective is to develop a practical automatic model that provides a useful level of automation while remaining understandable, maintainable, and retrainable in the Luminess environment. We choose an encoder–decoder architecture, as this type of architecture has shown strong performance in our state of the art.

Encoder

Based on the results of the state of the art, four encoder architectures with strong performance are compared: ResNet-50, Vision Transformer (ViT), Convolutional Vision Transformer (CvT), and Swin Transformer. These architectures represent different stages in the evolution from conventional convolutional architectures to modern Transformer-based approaches. This comparison allows us to evaluate whether more recent Transformer-based encoders provide a practical advantage for the MathML generation task.

The first encoder is an ImageNet-pretrained ResNet-50 [164], a convolutional neural network based on residual blocks. We use the first three layers of the network to preserve a more detailed spatial representation of the image, resulting in 1024 feature channels instead of 2048 with the final layer.

The second encoder is a Vision Transformer (ViT) [40]. It processes an image as a sequence of patches and uses self-attention between them. It is relevant for mathematical expressions, where dependencies can exist between visually distant elements such as fractions, superscripts, subscripts, and delimiters. The selected ViT configuration is the ViT-B/16 encoder in the default `vit_base_patch16_224` configuration with ImageNet-pretrained weights, an input resolution of 224×224 and a patch size of 16×16.

The third encoder is the Convolutional Vision Transformer (CvT) [42], which combines convolutional operations with Transformer-based self-attention. This design keeps the locality of convolutional networks while benefiting from the global modeling capabilities of Transformers. This architecture achieved great results on multi-font image to LaTeX recognition in the MathNet paper [58]. We use their architecture and pretrained encoder weights for our experiment, with an embedding dimension of 384, and input images rescaled from 300 DPI to the model required 600 DPI.

The fourth encoder is a Swin Transformer [41]. Swin divides the image into local windows and applies self-attention within each of them. By shifting the windows between consecutive layers, it can capture relationships across neighboring re-

gions while maintaining efficient computation. We use the pretrained Swin-Tiny `swin_tiny_patch4_window7_224` architecture and ImageNet weights for our experiment, with a default input resolution of 224×224, a patch size of 4×4, and a window size of 7. It provides a more recent Transformer-based alternative to the standard ViT architecture.

The main configurations used for the four encoders are summarized in Table 4.5.

Table 4.5: Encoder and decoder configurations used in the experiments.

Encoder				**Decoder**		
Architecture	**Pretrained**	**Output dim.**	d_{model}	**Layers**	**Heads**	**FF dim.**
ResNet-50	ImageNet	1024	2048	4	16	2048
ViT-B/16	ImageNet	768	768	4	16	2048
MathNet-CvT	im2latexv2	384	384	4	16	2048
Swin-Tiny	ImageNet	768	768	4	16	2048

Decoder and Tokenizer

The decoder is a Transformer decoder composed of 4 layers, with 16 attention heads and a hidden dimension that adapts to the encoder output dimension with an adaptive fully connected layer. It autoregressively predicts each token of the MathML sequence from the visual representations produced by the encoder.

The decoder generates the most probable token, one at a time. At each decoding step, the previously generated tokens are provided to the decoder together with the encoded visual representation. This process continues until the end-of-sequence token is produced. Positional encoding for the decoder uses sinusoidal embeddings, providing stable representations for variable-length sequences.

The output sequence is tokenized using the XML-aware tokenizer described in Section 4.2.3. This explicitly models the structure of the MathML document and reduces the risk of generating invalid XML tags. The vocabulary size is 352 tokens.

For reproducibility, the decoder configuration was adapted to the output dimensionality of each encoder, while keeping the remaining decoder parameters fixed across experiments. In all cases, the decoder uses four Transformer layers, 16 attention heads, a feed-forward dimension of 2048, a dropout rate of 0.01, and a maximum sequence length of 200.

Training Details

The models are trained using cross-entropy loss. The loss is computed over the predicted MathML tokens while ignoring padding tokens. We use the Adam optimizer with an initial learning rate of 10^{-4}. Each model is trained for a maximum of 20 epochs. The

best checkpoint is selected according to validation loss measured on the validation set. Training is performed on one NVIDIA A40 GPU.

Evaluation of MathNet Weights Initialization

The first experiments were conducted using the CvT encoder initialized with pretrained MathNet weights. However, this model produced unexpectedly poor results on the MathML generation task. The model achieved only 1.15% Exact Match and a normalized Levenshtein distance of 0.5567 after 20 epochs. Although the model was able to generate MathML sequences, the predictions contained many errors and hallucinations. To determine whether this poor performance was related to the CvT architecture or to the initialization, the same CvT architecture was trained from scratch. The results of the two configurations are presented in Table 4.6, and a few predictions from both models are shown in Figure 4.6.

Table 4.6: Comparison of the CvT trained from scratch and initialized with MathNet weights.

Model	Best epoch	Exact Match	Norm. Levenshtein
CvT from scratch	14	50.06%	0.0767
CvT MathNet	20	1.15%	0.5567

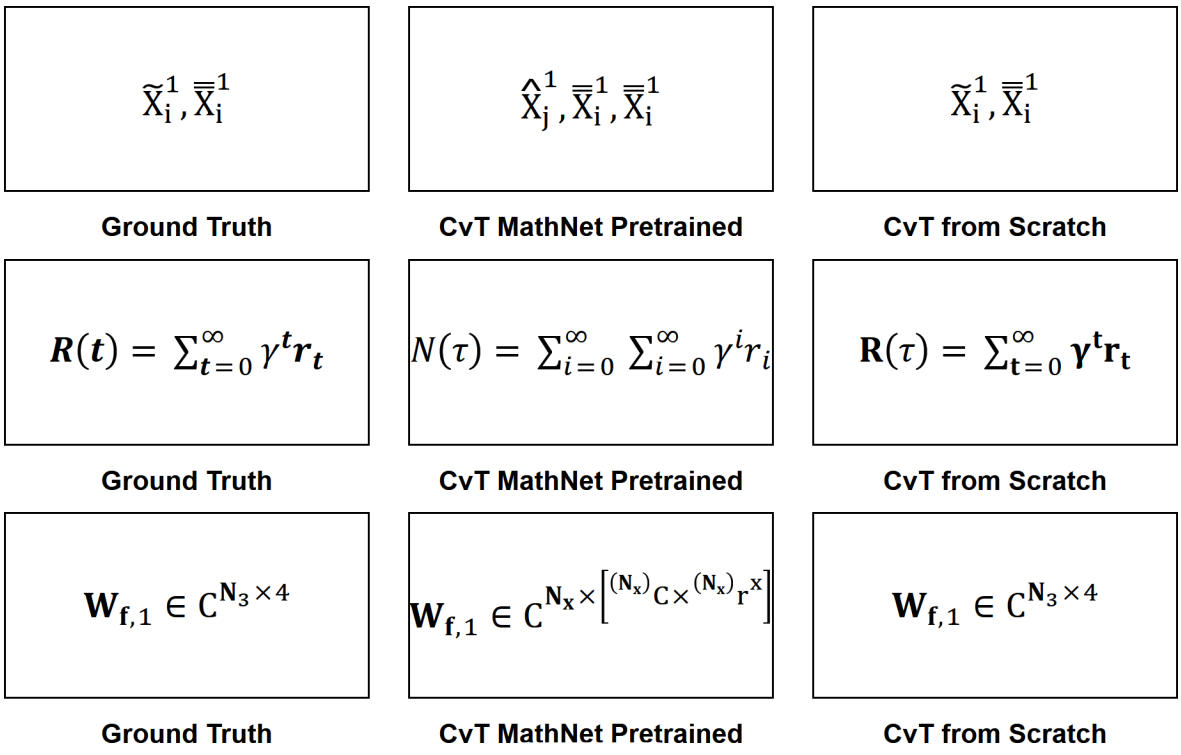


Figure 4.6: Examples of MathML predictions produced by the CvT initialized with MathNet weights and the CvT trained from scratch on the same input expressions.

When trained from scratch, the CvT reaches 50.06% Exact Match and a normalized Levenshtein distance of 0.0767. The large performance difference between the two configurations indicates that the poor results are not inherent to the CvT architecture and may be instead associated with the MathNet initialization.

However, this experiment does not allow the precise cause of the poor transferability to be isolated. MathNet was originally trained for image-to-LaTeX recognition, whereas MML-Net performs image-to-MathML generation. Even if the transferred parameters correspond only to the visual encoder, with the decoder being trained from scratch for the MathML vocabulary, the representations learned by MathNet may be specialized to the visual and structural characteristics relevant to LaTeX generation and may not transfer optimally to the MathML generation task.

Another difference concerns the image resolution used during training. MathNet was designed for images rendered at 600 DPI, whereas the images used in the present experiments originate from 300 DPI documents. To use the pretrained model, the 300 DPI images were upscaled by a factor of 2 to match the expected resolution. However, the MathNet paper reports that upscaling 300 DPI images performs worse than using images originally rendered at 600 DPI. The input images provided to the pretrained encoder do not fully match the conditions under which the MathNet visual representations were learned. This difference may further reduce the effectiveness of the transferred weights.

Overall, the results suggest that the MathNet initialization is not suitable for the proposed MathML generation task. For the comparison with the other encoder architectures, the CvT trained from scratch is kept as the CvT configuration in the next section.

4.3.2 Results of the Training Experiments

The selected architectures are trained on the PatentME-600k training data and evaluated on its validation set. The evaluation focuses on Exact Match and normalized MathML Levenshtein distance. Exact Match measures the proportion of predictions that are identical to the reference MathML sequence, and normalized Levenshtein distance measures the relative edit distance between the predicted and reference MathML.

Following the analysis of the MathNet initialization, the CvT trained from scratch is used for the comparison with the other encoder architectures. The results are presented in Table 4.7.

The results show a clear advantage of the Transformer-based encoders over the ResNet-50 baseline. The ViT obtains the highest Exact Match score, with 54.89%, followed by Swin-Tiny with 53.40% and the CvT trained from scratch with 50.06%. ResNet-50 obtains 39.70%. The same ranking is observed for normalized Levenshtein

Table 4.7: MML-Net results on the PatentME-600k validation set.

Encoder	Pretraining	Best epoch	Exact Match (↑)	Lev. Distance (↓)
ResNet-50	ImageNet	8	39.70%	0.1184
ViT	ImageNet	15	**54.89%**	**0.0631**
CvT	None	14	50.06%	0.0767
Swin-Tiny	ImageNet	19	53.40%	0.0688

distance. The ViT obtains the lowest distance, with 0.0631, followed by Swin-Tiny with 0.0688 and the CvT with 0.0767. ResNet-50 obtains a higher value of 0.1184.

The training behavior also provides information about the capacity of the different encoders.

ResNet-50 reaches its best validation performance earlier than the transformer configurations at epoch 8. In contrast, the Transformer-based architectures benefit from longer training, with the ViT reaching its best performance at epoch 15 and Swin-Tiny at epoch 19.

The Swin-Tiny results are particularly close to those of the ViT, with only a 1.49 percentage point difference in Exact Match and a 0.0057 difference in normalized Levenshtein distance. This suggests that Swin-Tiny may have potential for MathML generation and could potentially achieve higher accuracy, but at the cost of a longer training time, a larger configuration, and higher computational cost.

Overall, the ViT achieves the best performance on both evaluation metrics. We select it as the visual encoder for MML-Net in the remainder of this work.

4.4 Evaluation of MML-Net

This section evaluates the selected MML-Net ViT architecture against existing MER systems on several datasets and on PatentME-OCR, a dedicated test split of patent expressions. The evaluation uses the PiE-MER evaluation framework introduced in Chapter 3, which provides a common evaluation space for models generating different output representations.

4.4.1 Evaluation with PiE-MER

The evaluation is performed on four established LaTeX MER datasets, im2latex100k, im2latexv2, IBEM, and ME20k, as well as on PatentME-OCR. The four public datasets were introduced in the state-of-the-art chapter and provide standard benchmarks for MER. PatentME-OCR is generated using a process similar to PatentME-600k, gathering mathematical expressions extracted from European patent documents and annotated in MathML.

PatentME-OCR is used exclusively for evaluation. None of the evaluated models was trained using this dataset. This makes it possible to measure how existing MER systems generalize to the target patent domain and, in particular, whether a model trained directly on PatentME-600k benefits from domain-specific training.

PiE-MER converts the outputs of the different models into common representations. For models generating MathML directly and datasets of MathML expressions, a new *MathML mode* is implemented in PiE-MER, as illustrated in Figure 4.7. In this mode, the predicted and reference MathML expressions are first normalized using the parser described in Section 4.2.2. The normalized expressions are then rendered as images. In parallel, Symbol Label Graph representations [99] are generated to allow structural comparison.

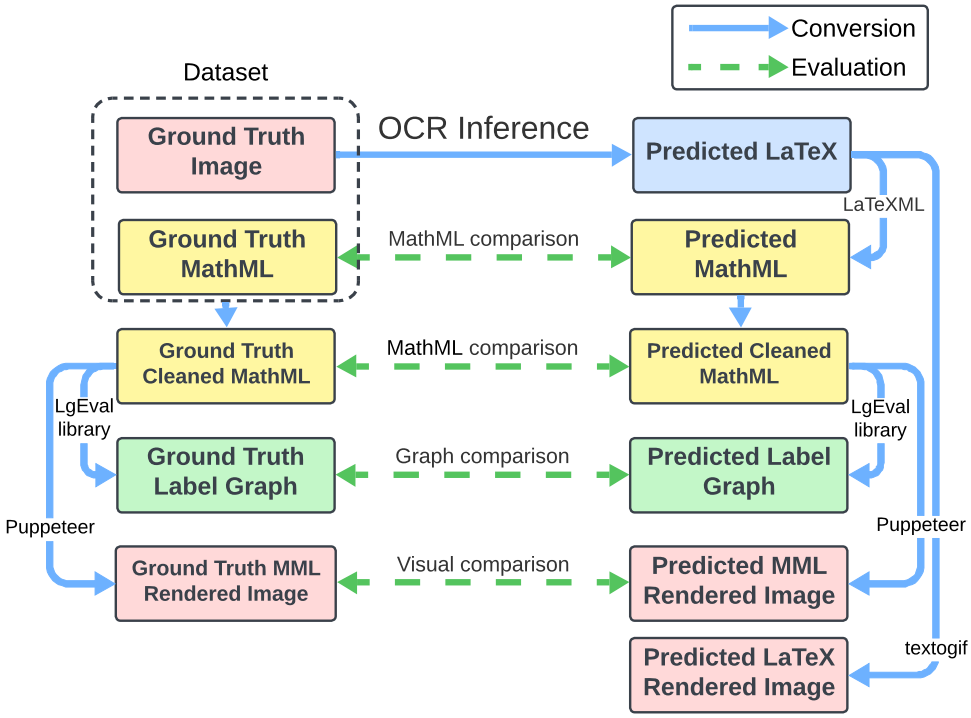


Figure 4.7: Conversion pipeline for the MathML mode of PiE-MER.

A MathML-to-LaTeX conversion is not performed because no sufficiently reliable conversion tool was identified for this evaluation. Instead, the rendered images and Symbol Label Graphs provide a common evaluation space for models producing LaTeX and models producing MathML.

For the four public datasets, both the reference expressions and the outputs of the evaluated LaTeX-based MER models are represented in LaTeX and rendered using LaTeX. For PatentME-OCR, the reference expressions and the outputs of MML-Net are represented in MathML and rendered using MathML. This avoids requiring a conversion between LaTeX and MathML and allows the different systems to be evaluated in their native output representation.

The main metrics are Visual Exact Match and Label Graph Exact Match. Visual Exact Match compares the rendered prediction with the rendered ground-truth and is equal to 1 only when the two images are identical pixel by pixel. For the public datasets and the reference models, the images are rendered from LaTeX, as LaTeX is their original output modality. For evaluation on PatentME-OCR and of MML-Net, the images are rendered from MathML, as MathML is the available markup representation. We refer to this metric as *Hybrid Visual Exact Match* because the same visual comparison protocol is applied across evaluation settings using images generated from different markup languages, namely LaTeX and MathML.

Label Graph Exact Match compares the corresponding Symbol Label Graph representations and evaluates whether the predicted graph is identical to the reference.

Table 4.8 summarizes the visual evaluation modalities according to the markup languages used by the dataset and the model output. To ensure a consistent visual comparison, the prediction and reference are first converted to a common markup language when necessary. The resulting expressions are then rendered from this common representation, and the rendered images are compared to compute the Visual Exact Match.

Table 4.8: Visual evaluation procedures according to dataset and model markup languages.

Model output modality	**Dataset reference modality**	**Evaluation procedure**
LaTeX	LaTeX	Render both expressions from LaTeX for visual comparison.
LaTeX	MathML	Convert the model output from LaTeX to MathML, then render both expressions from MathML for visual comparison.
MathML	LaTeX	Convert the dataset reference from LaTeX to MathML, then render both expressions from MathML for visual comparison.
MathML	MathML	Render both expressions from MathML for visual comparison.

These strict metrics are particularly appropriate for the target use case, as the objective is to produce expressions that can be directly integrated into the production pipeline. Even a small recognition error can have serious consequences for the patent publication process, as it could lead to the rejection of the entire document.

4.4.2 Compared Methods

Six models are compared in this evaluation. The selection includes dedicated MER systems, multimodal vision-language models, and MML-Net. We compare specialized mathematical OCR systems with more general-purpose models that have demonstrated strong performance on document and mathematical recognition tasks. They are summarized in Table 4.9.

Table 4.9: Models evaluated with PiE-MER.

Model	Architecture	Output	Training characteristics
TexTeller 2.0	Vision Transformer encoder–decoder	LaTeX	Trained on 7.5M diverse formulas
pix2tex	CNN–ViT encoder–decoder	LaTeX	Dedicated mathematical OCR
MathNet	Convolutional Vision Transformer encoder–decoder	LaTeX	Trained on multi-font im2latexv2
Qwen2-VL-OCR2-2B-Instruct	Vision-language model based on Qwen2-VL-2B	LaTeX	Fine-tuned for OCR and mathematical tasks
DotsOCR	NaViT vision encoder + Qwen2 language model	LaTeX	Large-scale multimodal pretraining and document OCR training
MML-Net	Transformer encoder–decoder	MathML	Trained on PatentME-600k

TexTeller 2.0 [165] is a Vision Transformer-based encoder–decoder MER model, trained on approximately 7.5 million diverse mathematical expressions. It is a large-scale specialized model designed specifically for MER and generates LaTeX representations of mathematical expressions.

pix2tex [166] is another dedicated mathematical OCR system based on a CNN–Vision Transformer encoder–decoder architecture. It also generates LaTeX sequences.

MathNet [58] is a specialized MER system based on a Convolutional Vision Transformer. It was trained on the multi-font im2latexv2 dataset and is designed to handle variations in mathematical fonts. MathNet expects input images to be scaled to 600 DPI, which is an important consideration when evaluating datasets acquired at lower resolutions. We rescale images to 600 DPI before running the model, as justified in Section 3.4.1.

In addition to these specialized MER systems, two recent vision-language models are included.

Qwen2-VL-OCR2-2B-Instruct [167] is a fine-tuned version of Qwen2-VL-2B-Instruct for OCR-related tasks, including mathematical recognition and LaTeX generation. Qwen2-VL is a multimodal architecture that combines a vision encoder with

the Qwen2 language model, allowing visual information to be processed jointly with language representations.

DotsOCR [76] is a document-oriented vision-language model designed for OCR and structured document understanding. Its architecture combines a NaViT-based vision encoder [168], which supports the processing of documents with varying resolutions and aspect ratios, with a Qwen2-based language model. Unlike dedicated MER systems, DotsOCR is trained as a general document OCR model rather than specifically for MER.

The inclusion of these two vision-language models is relevant because large multimodal models are exposed to very large and heterogeneous collections of documents, mathematical notation, and textual content during pretraining. They can acquire knowledge of mathematical notation and document layouts without being specifically designed as MER systems. In this evaluation, both models are prompted to generate LaTeX, as a large proportion of the mathematical content encountered during their training is likely to originate from scientific documents and PDF files in which LaTeX-based mathematical representations are common. It also provides a common output representation for comparison with the specialized MER systems. The prompts used for both models are provided in Appendix A.2.

Finally, **MML-Net** is the model proposed in this chapter. Unlike the other evaluated systems, MML-Net directly generates MathML and was trained on PatentME-600k, a large collection of mathematical expressions extracted from European patents. Its evaluation provides both a comparison with existing MER systems and clues on the potential benefit of domain-specific training for patent-oriented mathematical OCR.

4.4.3 Datasets

The evaluation uses five datasets: im2latex100k, im2latexv2, IBEM, ME20k, and PatentME-OCR. The first four are established LaTeX-based MER benchmarks introduced in the previous chapter. PatentME-OCR is generated in the same way as PatentME-600k, as it contains MathML markup and heterogeneous expressions extracted from European patents. We use it to evaluate generalization to our target patent domain. It contains 40,955 mathematical expression images extracted from 448 European patents, distinct from the one in PatentME-600k.

The distribution of image dimensions and MathML sequence lengths is shown in Figure 4.8. The dataset covers a broad range of image sizes and expression lengths, representing the heterogeneity of the patent expressions.

Examples of expressions from PatentME-OCR are presented in Figure 4.9. They have different sizes, noise levels, and fonts. Such variations are representative of the poor document conditions in patent applications and are not necessarily represented in

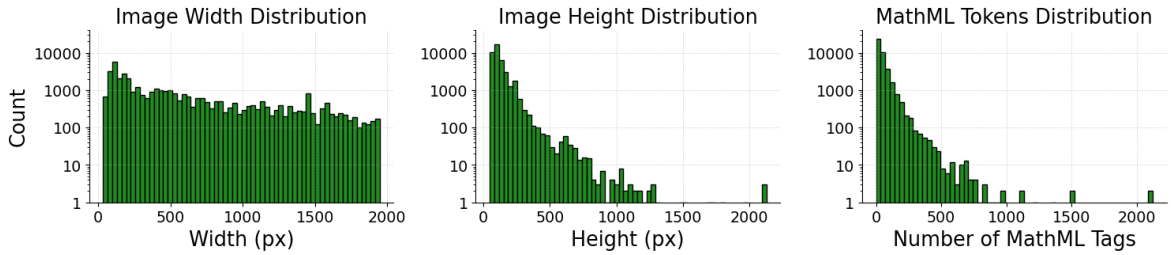


Figure 4.8: Distribution of image dimensions and MathML lengths in PatentME-OCR.

standard MER benchmarks.

$$P_{b,t} = P_m \cdot \eta_{total} = F_t v \cdot \eta_m \ \eta_{b,cha}$$

$$\left(\bigcirc_{f=R_2+1}^{R_1+R_2} Z_f \left(\left\{ z_j^{(F,f)} \right\}_{j=1}^{n_f} \right) \right)^{-1}$$

$$R_{ges}=1/(1/R_1+ 1/R_2+... 1/R_n),$$

$$1 < f \leq 2$$

Figure 4.9: Examples from PatentME-OCR with different sizes, qualities, and fonts.

4.4.4 Results and Discussion

The main results are presented in Figure 4.10, using heatmaps generated with PiE-MER. The first heatmap reports the Hybrid Visual Exact Match, while the second reports the Label Graph Exact Match .

The results show a clear separation between the public MER benchmarks and PatentME-OCR. No single model consistently performs best on the four public datasets, while all models experience a large performance drop on PatentME-OCR except MML-Net, which reaches 47.5% Visual Exact Match. The main results are discussed below by model family.

Specialized MER models

TexTeller 2.0 and pix2tex represent the behavior of specialized LaTeX-based MER systems. Their Label Graph Exact Match remains below 30% on im2latex100k and im2latexv2, but exceeds 60% on ME20k. The latter contains shorter and simpler high-school-level mathematical expressions collected from a website. This suggests that exact recognition remains difficult even on standard clean MER benchmarks based on scientific documents, while perfect recognition is more achievable for shorter and simpler expressions.

MathNet reaches 91.4% Visual Exact Match on the im2latexv2 test set, illustrating the importance of training and evaluating for a specific dataset, here with a focus on multi-font recognition. Its performance drops on ME20k. One possible explanation

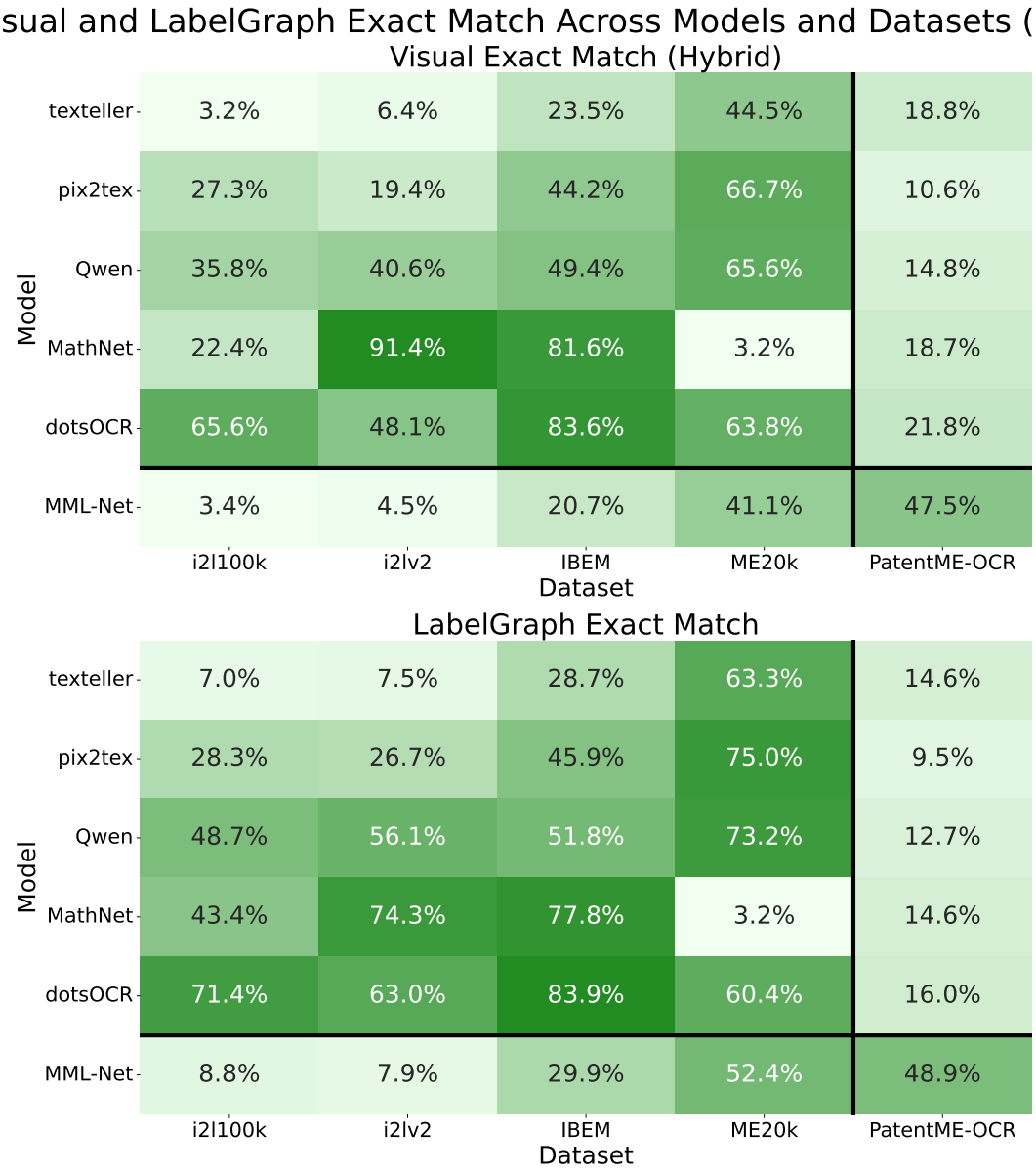


Figure 4.10: Hybrid Visual Exact Match of the evaluated models across the four LaTeX-based datasets and PatentME-OCR. The first four columns are evaluated from rendered LaTeX, whereas the PatentME-OCR column is evaluated from rendered MathML.

is the resolution of ME20k: MathNet expects 600 DPI inputs but ME20k contains 100 DPI images. Upscaling cannot recover the information lost during acquisition at 100 DPI and may introduce blur and interpolation artifacts. This result also shows that robustness to font variations does not transfer to robustness to changes in image resolution.

Vision-language models

The two VLMs achieve strong and relatively consistent results on the public benchmarks. Qwen2-VL-OCR2-2B-Instruct reaches between 35.8% and 65.6% Visual Exact Match, while DotsOCR reaches up to 83.6% on IBEM. These results show that general document-oriented VLMs can perform very well on mathematical expressions despite not being specifically designed for them.

Their strong performance may be explained by their large-scale pretraining on heterogeneous documents containing large amounts of mathematical and LaTeX content. However, these results should be interpreted with caution. The public test datasets used here may have been included in the training data. Possible benchmark leakage may contribute to their performance since the training sets are not fully disclosed.

However, this advantage does not transfer to PatentME-OCR. Qwen2-VL-OCR2-2B-Instruct reaches only 14.8% Visual Exact Match, while DotsOCR also drops to 21.8%. Strong benchmark performance does not imply equivalent robustness on unseen patent documents.

MML-Net

MML-Net exhibits the opposite behavior. Its performance on the public LaTeX-based datasets is generally low, except on ME20k, where it reaches 41.1% Visual Exact Match. It is not a general replacement for existing LaTeX-oriented MER systems. On PatentME-OCR, however, MML-Net reaches 47.5% Visual Exact Match and 48.9% Label Graph Exact Match, achieving the best results among all evaluated models. The latter indicates that the complete mathematical structure, including symbols and their relations, is correctly reconstructed for almost half of the patent expressions.

This result is expected because MML-Net was trained on PatentME-600k. The large gap between MML-Net and the general-purpose VLMs on PatentME-OCR suggests that large-scale multimodal pretraining alone may not be sufficient for this use case. Specific training on patent-domain mathematical expressions is required to obtain competitive performance. However, its performance remains far from sufficient for using it alone in a reliable industrial system. Patent MER is challenging even with domain-specific training.

Overall Discussion

The results show that public MER benchmarks provide only a limited view of specific patent recognition performance. Models that perform well on public datasets can perform poorly on PatentME-OCR because it contains patent-specific variations in style, image quality, and mathematical content. Our MML-Net model achieves reasonable performance and outperforms the other models on patent data, including large VLMs such as DotsOCR. This result highlights both the difficulty of the task and the benefit of training on domain-specific mathematical expressions.

The experiments also demonstrate the value of PiE-MER, which allows different kinds of models generating either LaTeX or MathML to be compared using the same visual and structural metrics despite their different output representations.

MML-Net was intentionally designed as a relatively simple architecture that is easy to understand and retrain. Its current performance can serve as a baseline for the rest of this thesis, while further optimization remains possible with larger models, improved architectures, longer training, and alternative training strategies.

4.5 Conclusion and Future Work

This chapter presented PatentME-600k and MML-Net, a dedicated dataset and MER model designed to generate MathML directly from expression images. PatentME-600k provides large-scale training data representative of mathematical expressions found in European patents, while MML-Net is designed for the target MathML representation.

Four visual encoders were evaluated: ResNet-50, ViT, CvT, and Swin-Tiny. The ViT achieved the best results. The final model reaches 54.89% Exact Match and a normalized Levenshtein distance of 0.0631 on the PatentME-600k validation set. The model is one of the first specialized image-to-MathML models and could help automate part of the mathematical transcription in a controlled processing pipeline, as demonstrated in Chapter 7.

The remaining gaps to improve recognition motivate the following chapters. Chapter 5 investigates reference-free post-OCR quality estimation, while Chapter 7 studies the prediction of recognition failures before recognition. It also proposes a semi-automatic processing chain in which MML-Net can operate in parallel with manual transcription, automatically processing expressions predicted to be suitable for OCR while difficult cases remain available for human transcription.

Several directions could further improve MML-Net and the PatentME dataset. First, with access to more GPU memory and computational resources, a larger portion of the EPO database could be processed using the dataset generation pipeline proposed in this work and made available at `github.com/fwieckowiak/PatentME`. This could

lead to larger versions of PatentME, similar to the successive im2latex datasets. For example, a future PatentME-2M dataset could contain several million mathematical expressions collected over a longer period.

Second, the current MML-Net could be improved with more recent encoder and decoder architectures, as well as alternative training strategies. Reinforcement learning could optimize a reward derived from Exact Match or structural similarity to better train the model. This could be particularly relevant for MML-Net, where a small number of token-level errors can cause an entire expression to be counted as incorrect under Exact Match.

Chapter 5

SiaEval : a Reference-Free Post-OCR Verification Task for Printed Mathematical Expression Recognition

5.1 Introduction and Motivation

For our use case of MER in patent applications, detecting potential recognition errors is essential to avoid mistakes propagating to the final documents and to downstream tasks of indexing and retrieval of patent literature. In conventional MER evaluation, the correctness of a prediction is measured against the textual ground-truth using metrics such as Exact Match, edit distance, or BLEU. These metrics require access to the reference markup and are unusable at inference time, where only the input expression image and the OCR prediction are available. This motivates a *reference-free* verification method that can estimate the likelihood that an OCR prediction is correct, without requiring the ground-truth markup.

An approach could be to compare the original expression image with an image rendered from the predicted markup. A clean visual representation of the prediction can be obtained simply by rendering it with a controlled rendering engine. However, it cannot be directly compared with the input image, as it may contain scanning noise, compression artifacts, degraded resolution, variations in font, character thickness, or spacing. Conversely, a small visual difference may correspond to a structurally significant recognition error, for example in a subscript, superscript, operator, or other small mathematical element.

This chapter proposes a solution to this problem through **SiaEval**, a reference-free post-OCR verification framework. It uses only the input expression image and the rendered OCR prediction image to determine whether the prediction is exact. To this end, a Siamese architecture is trained to learn a visual embedding space that is robust to variations in font, scale, and image quality while remaining sensitive to structural and semantic differences between mathematical expressions.

To determine whether SiaEval generalizes across different OCR models, it is first trained and evaluated on the errors produced by the TexTeller 2.0 MER model, an open-source MER model. A final section applies the framework to the predictions of MML-Net, our specialized MER model presented in Chapter 4. Since MML-Net produces much more subtle errors than TexTeller 2.0, this experiment also discusses the adaptations required to achieve satisfactory performance with this model.

5.2 Related Work

This section briefly reviews existing approaches to mathematical expression similarity and the use of Siamese Networks for visual similarity learning. We aim to position SiaEval with respect to existing similarity-based approaches and highlight the absence of suitable reference-free visual evaluation.

This chapter is based on results reported in [169].

5.2.1 Mathematical Expression Similarity and Retrieval

Mathematical expression similarity has been studied extensively in the context of mathematical information retrieval [170]. Existing approaches represent expressions using their textual or structural representation and compare them in a shared representation space. These representations may rely on the expression's semantic context, its tree structure, or learned embeddings of its textual representation. Other approaches rely on structural representations such as layout trees, recursive X–Y decomposition, or connected components [171, 143, 172, 173, 174, 175].

These approaches propose metrics to evaluate mathematical expression similarity but always require the textual markup of the reference expression. This allows the system to compare a query expression against a known representation of the reference. In the post-OCR verification scenario considered in this chapter, only the patent expression image and the rendered OCR prediction are available. No ground-truth markup is available at inference time, so these text or graph-based methods cannot be used.

Metrics and similarity scores based on rendered images or symbol layout graphs partially address this limitation [176]. Purely visual metrics such as IMEGE [177] compare rendered representations under controlled conditions.

Most of these approaches rely on a textual or controlled visual representation of the reference expression. However, the ground-truth markup is not available at inference in the post-OCR verification setting considered here, and the input image may contain scanning noise, compression artifacts, degraded resolution, variations in font, character thickness, or spacing. These limitations make direct textual or visual comparison impossible to use and motivate our reference-free approach.

5.2.2 Siamese Networks for Visual Similarity Learning

The problem is closely related to visual verification tasks in which two inputs must be compared despite variations in their appearance. Siamese Neural Networks (SNNs) are a natural framework for this type of problem [178]. They map two inputs into a shared embedding space and learn a distance function in which similar samples are close and dissimilar samples are separated.

Siamese architectures have been successfully applied to document and OCR-related tasks, including word spotting in historical Arabic manuscripts [179], symbol classification [180], Urdu character recognition [181], and recognition of low-quality document text [182].

They have also been applied specifically to mathematical expressions. For example, [183] learns visual embeddings of mathematical expressions for retrieval and highlights the importance of hard negative sampling. In handwritten MER, [184] combines contrastive learning with an OCR objective using paired handwritten and rendered

expressions. These approaches use visual embeddings either for mathematical expression retrieval or as part of the recognition process. They do not address the problem of verifying whether a generated mathematical markup correctly represents a given input image without access to its ground-truth transcription.

5.2.3 Reference-Free Evaluation with LLMs

Recent works have investigated the ability of LLMs to measure the reliability of their own outputs or to evaluate outputs produced by other systems. In [185], the authors study whether the confidence expressed by a GPT model is calibrated with the actual quality of its predictions. Similarly, [186] use LLMs as evaluators to estimate the quality of named entity recognition and linking outputs in a reference-free setting, obtaining promising results in noisy conditions.

For MER, [187] analyze the entropy of the predictions generated by a ChatGPT-based MER system and show relationships between this entropy, recognition performance, and the quality and resolution of the input image. LLM-based evaluation provides a natural reference-free baseline for this kind of setting because the model can directly inspect the input and the predicted output without having access to ground-truth annotations.

However, these approaches primarily rely on the language model's confidence or reasoning capabilities rather than on an explicit visual similarity between the input expression and the predicted output.

5.2.4 Positioning of SiaEval

The SiaEval architecture proposed in this chapter combines the reference-free setting of post-OCR verification with the visual similarity learning of Siamese Networks. Unlike conventional MER evaluation and most mathematical expression similarity approaches, it does not require the ground-truth markup at inference time. Instead, it learns from pairs of original expression images and rendered OCR predictions whether the two visual representations correspond to the same mathematical expression.

To the best of our knowledge, no previous work has formulated post-OCR verification of mathematical expressions as a visual similarity problem between an original expression image and its rendered OCR prediction. This chapter introduces this formulation and investigates the use of a Siamese architecture to determine whether the generated prediction correctly represents the input expression.

5.3 Methodology

5.3.1 Problem Formulation

In this section, a lowercase letter represents a MathML markup string, while an uppercase letter denotes its rendered image. Let a reference image of a mathematical expression X be associated with a correct MathML representation y^+ and an incorrect MathML representation y^-, together with their corresponding visual renderings Y^+ and Y^- (Figure 5.1).

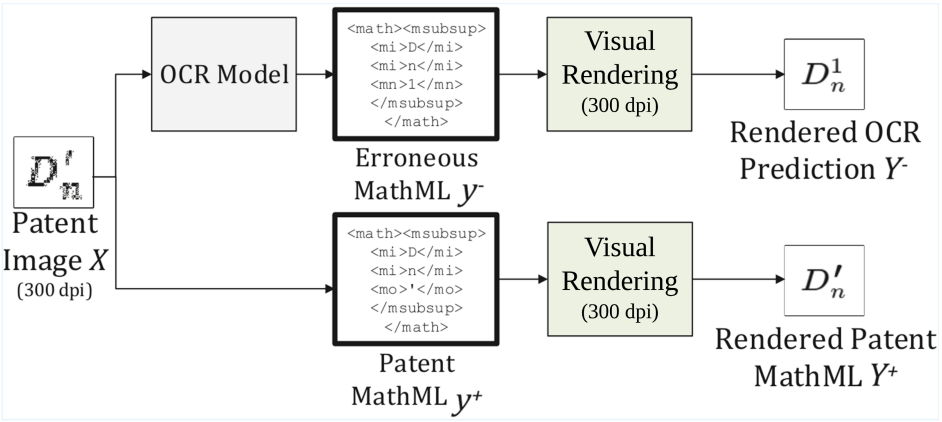


Figure 5.1: Generation pipeline of the reference image X, positive image Y^+, and negative image Y^-.

Our objective is to train a model that, given the reference image X and a candidate image $\hat{Y}$, can determine whether the candidate image corresponds to a correct or incorrect MathML markup string. This task is not trivial because the reference image X can have arbitrary font, quality, and style. Direct comparison between X and $\hat{Y}$ using simple image similarity metrics is not sufficient.

To address this problem, a Siamese Neural Network, illustrated in Figure 5.2, is used to embed both images into a shared embedding space and compute the distance between the resulting vectors. Let $\mathbf{e}_X$ and $\mathbf{e}_{\hat{Y}}$ denote the embeddings of the two expressions. Their similarity is measured using the angular distance. For very close vectors, cosine distance compresses small differences, whereas angular distance is linear with respect to θ, providing finer discrimination between highly similar embeddings. Early experiments also showed that angular distance was more effective for this task. It is defined as:

$$d(\mathbf{e}_X, \mathbf{e}_{\hat{Y}}) = \frac{1}{\pi} \arccos\left(\frac{\langle \mathbf{e}_X, \mathbf{e}_{\hat{Y}} \rangle}{\|\mathbf{e}_X\| \, \|\mathbf{e}_{\hat{Y}}\|} \right), \qquad d \in [0, 1].$$

Using a contrastive loss, the model is trained to keep the embeddings of X and Y^+ close while pushing the embeddings of X and Y^- apart. A contrastive loss is preferred over a triplet loss because it directly optimizes the distance between positive

and negative pairs, resulting in an interpretable similarity measure. In contrast, triplet loss enforces a relative ordering between an anchor, a positive, and a negative sample, which is less directly aligned with the verification task. Triplet loss will be used for a mathematical expression sorting task in Chapter 6. The contrastive loss is defined as:

$$\mathcal{L}(X, \hat{Y}, z) = z\, d(\mathbf{e}_X, \mathbf{e}_{\hat{Y}})^2 + (1 - z)\, \max\Big(0, m - d(\mathbf{e}_X, \mathbf{e}_{\hat{Y}})\Big)^2,$$

$$\text{with } z = \begin{cases} 1, & \text{if } \hat{Y} = Y^+ \\ 0, & \text{if } \hat{Y} = Y^- \end{cases}, \text{ and } m > 0 \text{ is a margin hyperparameter.}$$

The classification decision $\hat{z}$ is obtained by thresholding the distance:

$$\hat{z} = \begin{cases} 1, & \text{if } d(\mathbf{e}_X, \mathbf{e}_{\hat{Y}}) \leq \tau \\ 0, & \text{otherwise,} \end{cases} \qquad \text{with } \tau \text{ the selected similarity threshold.}$$

In this binary classification problem, a false positive corresponds to an incorrect prediction incorrectly flagged as correct, while a false negative corresponds to a correct prediction incorrectly flagged as erroneous. For our high-precision scenarios, the decision threshold τ is selected on a validation set using a Precision–Recall curve to enforce a very low number of false positives. Positive pairs correspond to images representing the same formula, while negative pairs correspond to prediction errors. The threshold τ is selected to achieve at least 95% precision on the positive class. This ensures that pairs predicted as identical are almost always correct (high precision) and limits the number of negative pairs incorrectly classified as positive (false positive). With a selected threshold, model performance is evaluated using recall on the positive class, i.e. the ratio of identified positive samples over all actually positive samples. Having some false negative pairs is less critical in this setting, as they can be corrected by downstream models and manual verification.

To ensure that the model is not overly conservative at this high-precision operating point and rejects all the expressions, recall is measured at 95% precision. A low recall would indicate that the model accepts only a small proportion of the correct predictions as positive, and is very conservative. Conversely, a high recall at 95% precision indicates that the model can automatically accept a larger proportion of correct predictions while maintaining the required precision.

5.3.2 Data Acquisition

Training the Siamese network requires reference mathematical expressions with heterogeneous image quality, their ground-truth markup, and incorrect OCR predictions. To this end, we rely on the PatentME-OCR dataset introduced in Section 4.4.3 and on

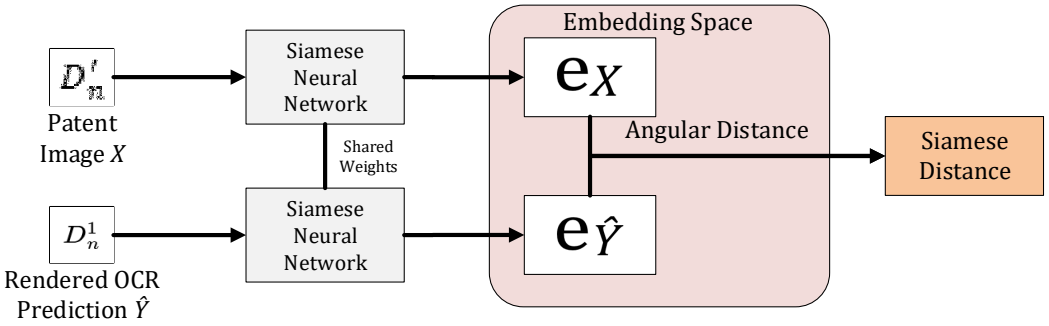


Figure 5.2: The SNN architecture for the verification task. The reference and predicted images are encoded by a shared network and compared in the embedding space.

the evaluation results presented in Section 4.4.4. To demonstrate that the proposed framework can be applied to an existing OCR system without requiring modifications to the recognition model itself, the predictions of TexTeller 2.0 are used to construct the training data.

As a reminder, PatentME-OCR consists of 40,955 mathematical expression images extracted from 448 European patents, together with their corresponding MathML representations extracted from the patent XML files. MathML simplification and normalization were applied to ensure reliable conversion and rendering without altering the visual appearance or meaning of the expressions (see Section 4.2.2). Tags such as `<mstyle>`, `<mpadded>`, and `<mspace>` were replaced because they interfere with the rendering pipeline, while the deprecated `<mfenced>` tag was replaced with an `<mrow>` containing explicit delimiters and separators. Figure 5.3 illustrates the variability in image quality, style, and font present in the dataset, making post-OCR verification particularly challenging.

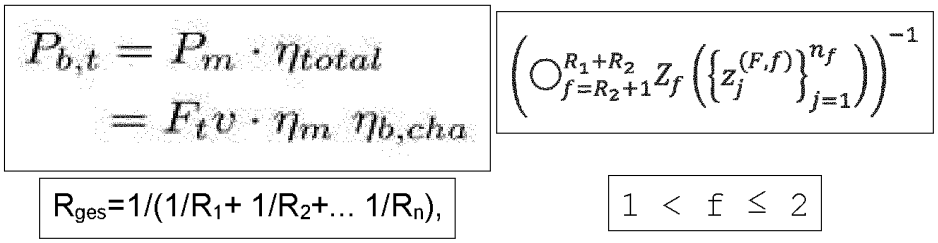


Figure 5.3: Examples of mathematical expressions from PatentME-OCR with different sizes, qualities, and fonts.

The predictions of TexTeller 2.0 [165] are collected on the PatentME-OCR dataset introduced in Section 4.4.3. TexTeller 2.0 is a MER model trained on 7.5M diverse mathematical expressions and designed to handle rare symbols, complex multi-line expressions, and matrices. The reference images X, ground-truth markup y^+, rendered ground-truth images Y^+, predicted markup $\hat{y}$, and rendered predictions $\hat{Y}$ are extracted from the evaluation conducted in Section 4.4.4.

These outputs are then used to construct positive and negative pairs, associating each reference image with either a correct or an incorrect rendered prediction. The resulting dataset consists of the positive and negative pairs (X, Y^+) and (X, Y^-) defined in Section 5.3.1. Positive samples Y^+ are generated by rendering the ground-truth MathML extracted from the patent XML, while negative samples Y^- are generated by rendering incorrect MathML predictions produced by TexTeller 2.0 (see Figure 5.1).

This procedure results in the PatentME-Siamese dataset, consisting of 67,680 pairs, evenly split between positive (33,840) and negative (33,840) samples. Each sample contains the reference patent image X and the query image $\hat{Y}$, a binary label, and the metadata associated with the model evaluation. The dataset is split into 60,972 training pairs, 3,346 validation pairs, and 3,362 test pairs, ensuring the coresponding positive and negative samples are in the same split.

Using the errors produced by a single OCR model exposes the verifier to realistic prediction errors from a specific recognition system, rather than relying on artificially generated or overly obvious negative examples. Adapting SiaEval to another OCR model or recognition domain would require retraining the verifier on representative errors of the target system, as done in Section 5.6.

5.3.3 Siamese Network Architecture

Our Siamese pipeline is illustrated in Figure 5.2. We select a lightweight ResNet-18 CNN as the encoder, which has proven effective for mathematical expression embedding [158] and allows for rapid experimentation. A fast and compact architecture is also desirable for real-time post-OCR verification. The network is trained from scratch, as the size of the dataset is sufficient to learn a task-specific representation.

The contrastive loss and angular distance described in Section 5.3.1 are used for training and comparison, respectively. The contrastive margin m is treated as a hyperparameter to be optimized. To determine an operating point for discriminating between positive and negative decisions based on the distance produced by the network, an optimal threshold τ is selected on the validation set to achieve a minimum precision of 95%. This threshold can be determined from the Precision–Recall curve. The general performance of the model across operating points is also quantified by the area under this curve, measured using Average Precision (AP).

The selected threshold is then applied to the test set to obtain the final positive and negative predictions. These predictions are compared with the ground-truth labels to produce the confusion matrix, from which the final test precision and recall are computed.

We also perform a study on the impact of input image preprocessing. For the ResNet-18 SNN, we resize all input images to a fixed square size of either 224^2 or 512^2

pixels, and we compare two rescaling strategies: resizing while preserving the aspect ratio with top/bottom or left/right padding (Padding), and rescaling to fully fill the square input size (Scaling). These preprocessing methods are illustrated in Figure 5.4.

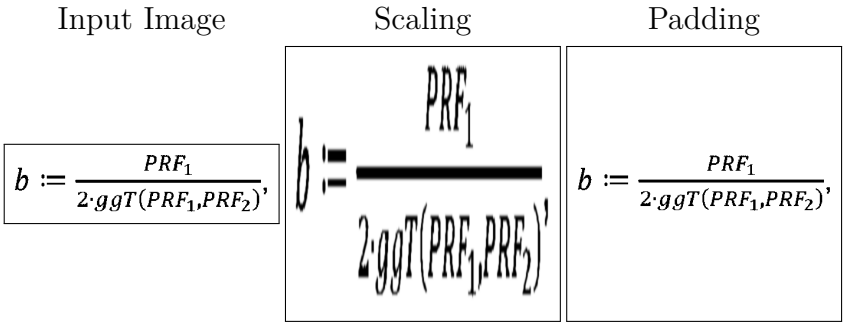


Figure 5.4: The two preprocessing methods evaluated in our study.

We optimize the margin m through Bayesian optimization over 14 trials on the validation set, each trained for 20 epochs, with the same objective of maximizing recall @95% precision. The final model with the best input preprocessing and margin is then trained and evaluated 7 times for 50 epochs. We use a batch size of 128 and the Adam optimizer with a cosine-annealing learning rate scheduler to improve convergence. We report the losses, average precision, and test recall over time, as well as the validation Precision-Recall curve and test confusion matrix of the best model on its best epoch.

5.3.4 GPT-5-mini Baseline

We use GPT-5-mini as a zero-shot VLM baseline because it can directly compare the reference image and the rendered OCR prediction without the ground-truth transcription at inference time. We compare it against our Siamese approach. The prompt in Figure 5.5 instructs the model to determine whether the reference and query images represent the same or different expression.

System prompt

You are an expert AI system for rigorous mathematical expression comparison.
Task: Given two images, determine whether the represented mathematical expressions are semantically identical.
Ignore differences in font, size, and equation numbering.
Bold and italic carry semantic meaning and must be treated as differences.
Layout differences are significant: a stacked (vertical) fraction and an inline fraction (e.g., 1/2) must be considered different.
Output strictly a JSON object with the following field:
{distance: a float in [0,1]}. The distance value must satisfy: 0 indicates identical expressions, 1 indicates completely different expressions.

Figure 5.5: Prompt used with GPT-5-mini

To limit the token usage and cost of this experiment, we scale down the images such that their longest side does not exceed 512 pixels while preserving aspect ratio. Like the Siamese model, GPT-5-mini returns a distance value that represents how similar the two expressions X and $\hat{Y}$ are. We use the same pipeline to find τ on the validation set and evaluate on the test set. GPT-5-mini is prompted to provide a numerical similarity distance in the range [0,1], allowing us to apply the same threshold selection and evaluation procedure to both approaches.

5.4 Results

5.4.1 Siamese Model

Table 5.1 reports the impact of input resolution (224^2 vs. 512^2) and preprocessing method (Padding vs. Scaling) on the Siamese network described in Section 5.3.3, trained for 50 epochs with margin $m = 0.5$. Padding consistently yields higher test recall than scaling at both resolutions, with the best performance obtained using 512^2 inputs with padding with a best recall of 72.99% at epoch 41. Models trained with 224^2 inputs reach peak recall earlier (epochs 27–28), indicating early convergence and limited gains from additional training. Bayesian optimization of the margin parameter m was conducted over 14 trials using 512^2 inputs with padding. An optimal margin of $m = 0.33$ was found for our architecture.

Table 5.1: Impact of input preprocessing on the validation recall at 95% validation precision. Training over 50 epochs.

Input Processing	Final Recall (%)	Best Recall (%)	Epoch of Best
Padding 224^2	59.49	62.64	28
Scaling 224^2	43.19	55.68	27
Padding 512^2	**68.71**	**72.99**	41
Scaling 512^2	27.36	31.23	36

Figure 5.6 reports the performance of the final model (margin $m = 0.33$, 512^2 inputs with padding) over 7 runs. It shows the average training and validation losses, validation average precision, and test recall across 50 epochs.

Figure 5.7 reports the evaluation of the best run, including the validation precision-recall curve, test confusion matrix, and predicted test distance distributions. With a decision threshold $\tau = 0.068$ selected to ensure at least 95% validation precision, the model achieves a true negative rate of 95.30% and a recall of 77.81% on positive pairs in the test set. The distributions of distances for positive and negative samples are well separated with some overlap. The decision threshold $\tau = 0.068$ separates the two groups in the embedding space and limits the number of false positives. The observed angular

distances lie in [0, 0.5] rather than [0,1] due to the embedding optimization: vectors for positive pairs are close (the distance is close to 0) and negatives are separated just enough to satisfy the contrastive margin of $m = 0.33$.

Manual verification of the false positives and false negatives produced by the Siamese model was conducted to confirm the ground-truth labels. Some predictions initially labeled as errors were found to be correct upon manual review. This indicates that there are some minor inconsistencies in the dataset. Out of 373 false negatives, 29 were corrected to true negatives, and 12 out of 79 false positives were corrected to true positives. Overall, the corrected samples represent 1.22% of the entire test set.

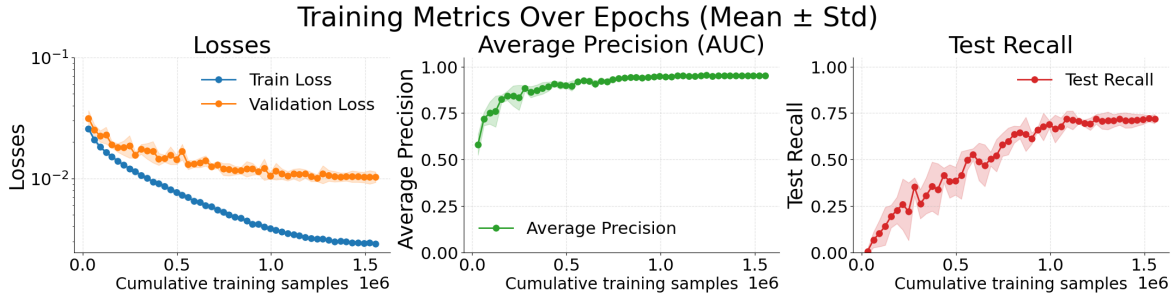


Figure 5.6: Evolution of the training metrics for the final SNN model (margin $m = 0.33$, 512^2 input with padding) averaged over 7 runs.

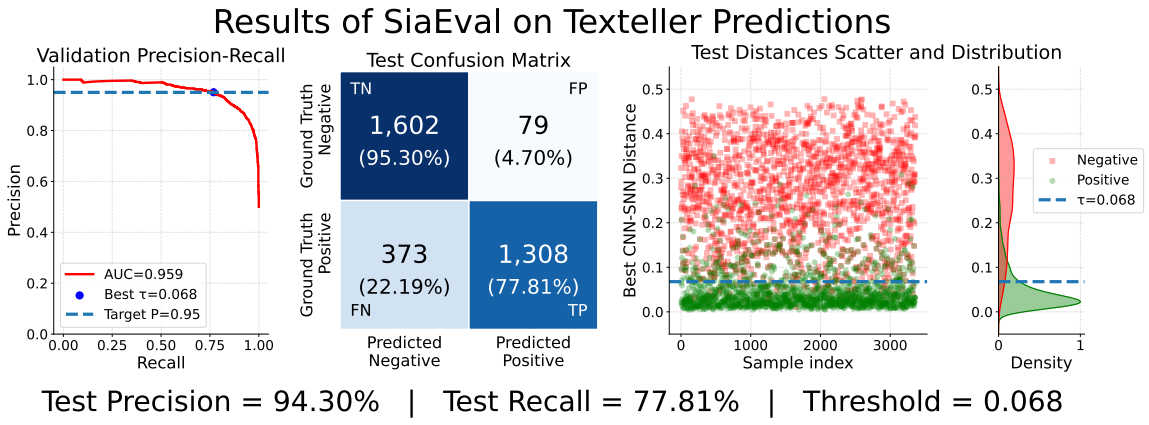


Figure 5.7: Validation precision-recall curve, test confusion matrix, and distribution of predicted distances on the test set for the best SNN model.

5.4.2 GPT-5-mini Approach

The same protocol was applied to the GPT-5-mini baseline with a threshold selected on the validation set to ensure at least 95% precision. In practice, only $\tau = 0$ satisfied this requirement, labeling all samples as negative, resulting in a 0% recall. To circumvent this, the test threshold was set to $\tau = 0.00^{+}$, meaning that only samples with a distance exactly equal to 0 are labeled as positive, while any sample with a distance strictly greater than 0 is labeled as negative.

At this operating point on the test set, the model achieves a true negative rate of 91.02% and a recall of 38.96% on positive pairs. Figure 5.8 shows the PR curve, test confusion matrix, and distance distributions for this baseline. Compared to the Siamese model, GPT-5-mini shows lower verification performance: its recall is much lower (38.96% vs 77.81%) even with a slightly lower true negative rate (91.02% vs 95.30%). The distribution plot shows distances clustered near 0 and 1, reflecting a sharp separation between inputs perceived as similar versus different. However, many negative samples appear near 0, where ideally only positives should be located. This overlap corresponds to potential false positives and explains why a very strict test threshold of $\tau = 0.00^+$ was required to maintain the desired precision.

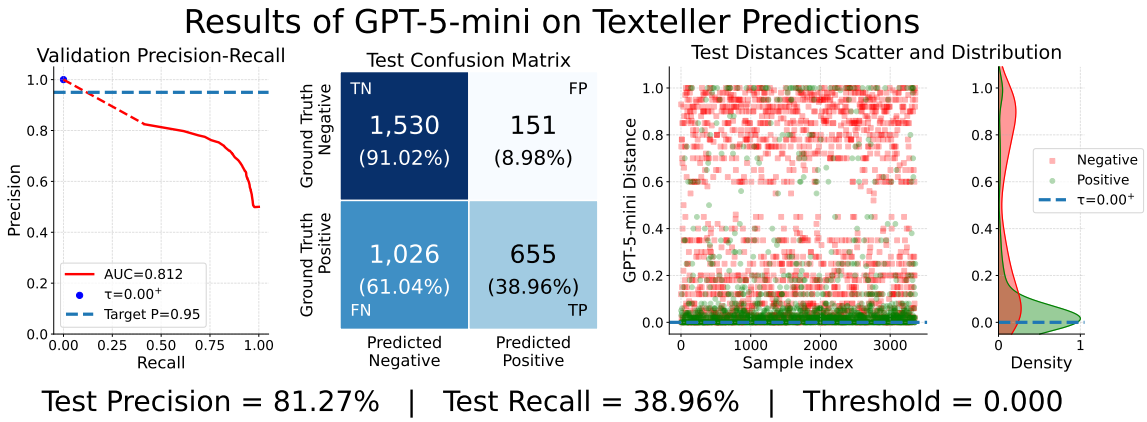


Figure 5.8: Validation precision-recall curve, test confusion matrix, and visualization of predicted distances between samples on the test set for the GPT-5-mini baseline.

5.5 Discussion

5.5.1 Impact of input processing

The input resolution analysis shows that using a small input size leads to a drop in performance. Equations in PatentME-OCR vary between 0 and 2000 pixels in width, and most of them are under 512 pixels in height. Reducing them to 224^2 compresses fine details and visual information. A 512^2 input window preserves enough details for accurate recognition at the cost of a higher GPU usage and longer training time.

For the rescaling strategies, padding consistently outperformed scaling. This may be due to the fact that the task is sensitive to symbol shapes and relative positioning. Scaling distorts these properties, while padding preserves them even if it introduces white regions. Also, some patent expressions include equation numbers in the images that must be ignored in the verification task. They can artificially increase the image size and force aggressive down-scaling to fit the input window, degrading performance.

Identifying them and recropping the images may help solve this issue, at the risk of removing some relevant parts of the expressions.

5.5.2 Comparison with the GPT-5-mini Baseline

Our simple ResNet-18 SNN baseline achieves strong performance despite its simplicity and limited training set of 61k samples. Even under these constraints, it outperforms the GPT-based baseline on the PatentME-Siamese comparison task. Vision Transformer-based encoders could likely capture even finer visual differences and further improve performance. The naive GPT-5-mini approach performs worse, likely because it is not optimized for the specific comparison task. These results show the benefit of task-specific visual metric learning over a naive strategy for this verification task.

The main source of errors appears to be image quality rather than expression complexity. Qualitative observations show that both simple and complex equations can be correctly matched, suggesting that the model is robust to variations in complexity. Many errors are instead caused by extremely low-quality patent scans and very large expressions that must be scaled down to fit the CNN input size. This downscaling can cause fine details to be lost and makes errors more difficult to detect.

Some other limitations include the quality of the annotations used to generate the positive and negative pairs and the presence of mislabeled samples in the source XML data. Manual verification of the false positives and false negatives in the test set revealed that 1.22% of the samples were mislabeled, mostly due to errors in the patent XML files. Since the training and validation sets were not manually verified, similar inconsistencies may also be present in these splits. This label noise is a limitation of the available data and may affect the reliability of the learned metric.

The current analysis of error causes is also mainly qualitative; a quantitative analysis of the factors affecting verification errors, such as image quality, expression size, and error type, would provide a better understanding of the model's limitations. In addition, the 512^2 input resolution can lead to a loss of fine details in very large expressions. Larger or more specialized encoders, as well as evaluation on additional patent data, could further improve the results.

These results show that a lightweight Siamese network can provide an effective reference-free verification for MER predictions. The results demonstrate the feasibility of learning a visual verification metric from OCR errors. The next section attempts to transfer this model to verify the predictions of MML-Net, whose errors are considerably more subtle than those produced by TexTeller 2.0.

5.6 Adaptation to MML-Net

The previous experiments demonstrated the SiaEval framework on the errors produced by an existing MER model, TexTeller 2.0. We refer to this version of the model as **SiaEval-TT**. The next objective is to apply the same verification pipeline to the predictions of MML-Net, the model presented in Chapter 4, which is intended to be integrated into the final recognition pipeline.

MML-Net achieves higher recognition performance on the same PatentME-OCR dataset, with a Visual Exact Match rate of 47.5%, compared with 18.8% for TexTeller 2.0. Its remaining errors are expected to be more difficult to detect. Two experiments are conducted to investigate this difference. First, SiaEval-TT is directly applied to the MML-Net predictions without any retraining. Second, a new SiaEval model is trained specifically on MML-Net errors, referred to as **SiaEval-MN**, using the same overall framework but with an adapted image encoder.

5.6.1 Transfer of SiaEval-TT

SiaEval-TT is first applied directly to the validation MML-Net predictions on PatentME-OCR. The resulting distances are shown in Figure 5.9. The model fails to provide a useful operating point at 95% precision for distinguishing correct and incorrect MML-Net predictions. The distributions of distances for the two classes overlap completely, preventing the selection of a threshold that provides any Precision–Recall trade-off.

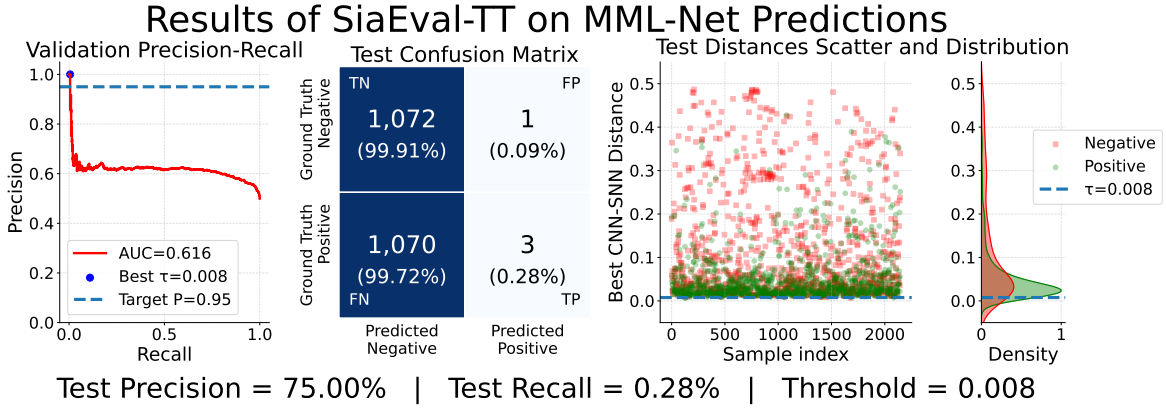


Figure 5.9: Application of SiaEval-TT, trained on TexTeller 2.0 errors, to MML-Net predictions.

This result shows that the representation learned by SiaEval-TT does not transfer to the error distribution of MML-Net. MML-Net errors appear to be much harder to distinguish visually than those produced by TexTeller.

This difference can be observed by comparing the distributions of the normalized edit scores of the two OCR models, shown in Figure 5.10. Most MML-Net predictions

have edit scores concentrated between 0.9 and 1, indicating that many of its predictions involve only small differences from the correct expression. In contrast, TexTeller 2.0 errors are more broadly distributed, with a larger proportion occurring between approximately 0.6 and 0.9. These errors correspond to more visible differences between the predicted and correct expressions and are easier for SiaEval-TT to distinguish.

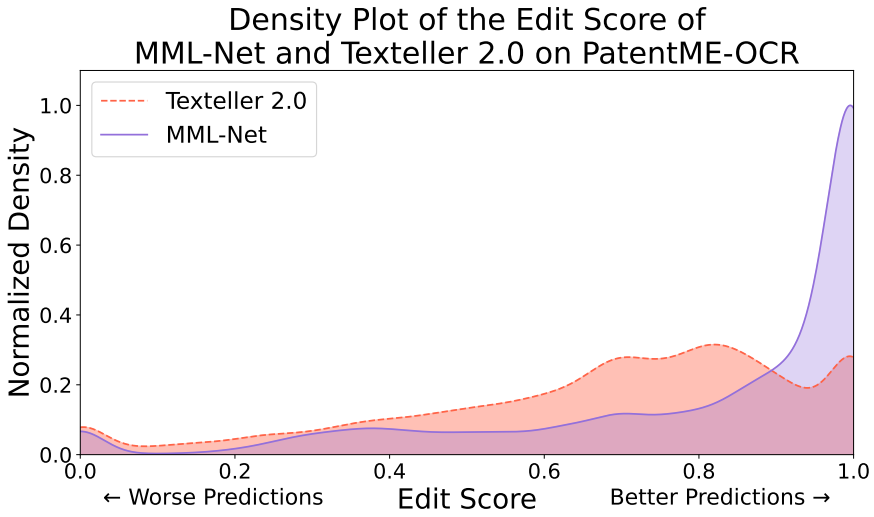


Figure 5.10: Distribution of normalized edit scores for TexTeller 2.0 and MML-Net predictions on PatentME-OCR.

Examples of these subtle MML-Net errors are presented in Figure 5.11, together with the corresponding reference patent image and TexTeller 2.0 prediction. In each case, the MML-Net prediction differs from the reference by only a small visual detail, such as a single symbol, character, or local structural element. Such differences are harder to detect than the larger errors produced by TexTeller 2.0.

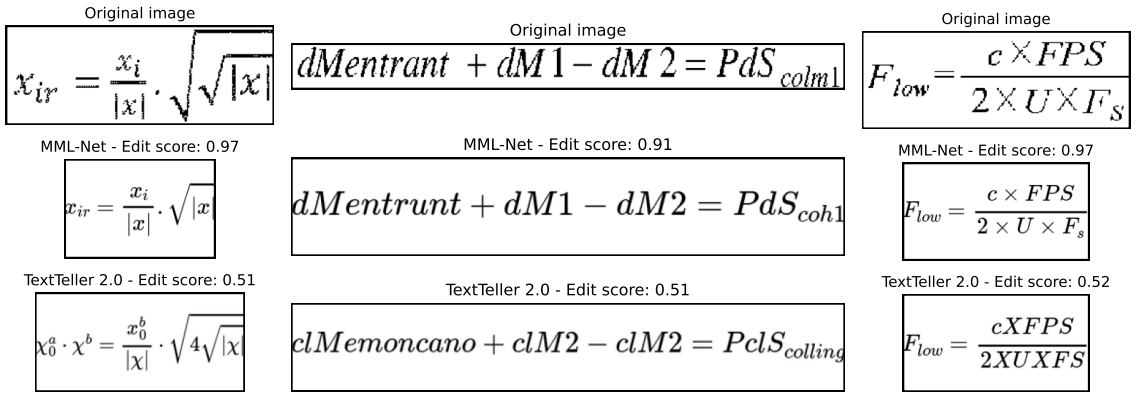


Figure 5.11: Examples of subtle and more obvious recognition errors produced by MML-Net and TexTeller 2.0.

These observations explain the poor transfer performance of SiaEval-TT. The model has learned a representation adapted to the visual characteristics of TexTeller 2.0 errors, while MML-Net produces a different and more challenging error distribution.

5.6.2 SiaEval-MN

To adapt SiaEval to these harder samples, we modify some aspects of the visual encoder. The original model uses a ResNet-18 encoder trained from scratch with a 512×512 input window. The new encoder instead uses ImageNet-pretrained weights and a larger rectangular input of 1024×320 pixels. The ImageNet pretraining provides a more informative initialization than training the encoder from scratch, while the new input resolution increases the input surface area by 25% and better matches the wide rectangular shape of mathematical expressions in PatentME-OCR. The image is resized to fit in the window while preserving its aspect ratio, then centered and padded to obtain the 1024×320 input window. Compared with the original 512^2 input window, this reduces the quantity of white padding above and below the expression and should preserve more visual details that may be important to distinguish correct expressions from subtle OCR errors.

The same hyperparameters defined in Section 5.3.3 are used to train SiaEval-MN. This adaptation changes only the visual representation and input preprocessing, while the data acquisition, verification principle, loss function, and decision procedure remain unchanged.

SiaEval-MN results are presented in Figure 5.12. At the operating point selected to maintain 95% validation precision, SiaEval-MN uses a threshold $\tau = 0.057$ and reaches a test true negative rate of 97.30% and a recall of 59.37% on the positive class. This means that 97.30% of the erroneous predictions are correctly flagged as erroneous, while 59.37% of the actually correct predictions are flagged as correct. This is a large improvement over the direct application of SiaEval-TT and demonstrates that quickly retraining the verifier on representative errors from the target OCR model is effective.

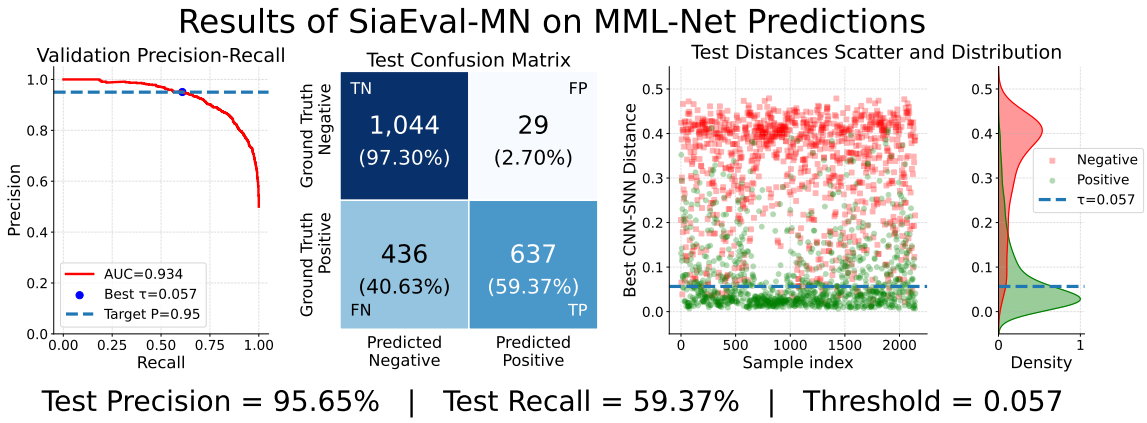


Figure 5.12: Performance of SiaEval-MN trained on MML-Net errors.

The lower recall, compared to the results of Section 5.4 which reached 78%, reflects the increased difficulty of the MML-Net verification task. The threshold can be adjusted depending on the requirements of the downstream application. For example, targeting

a lower 90% validation precision results in a more lenient threshold of $\tau = 0.098$, with a lower true negative rate of 92.45% but a recall of 73.72%. This operating point provides a recall closer to that obtained previously, at the cost of accepting a higher proportion of false positives. This second operating point is displayed in Figure 5.13.

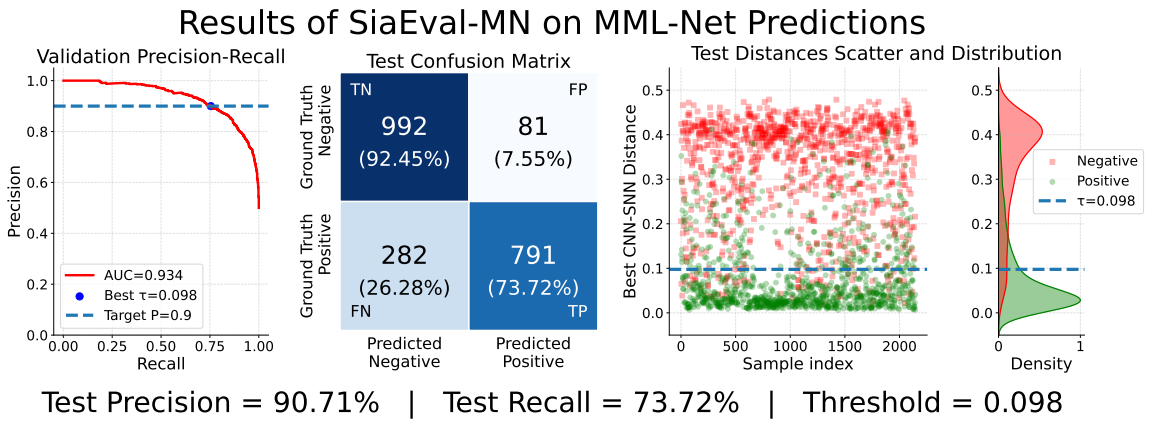


Figure 5.13: Performance of SiaEval-MN trained on MML-Net errors at a more lenient operating point.

5.6.3 Ablation Study

To measure the contribution of each adaptation introduced for MML-Net, an ablation study is conducted by progressively modifying SiaEval-TT. The experiments start from the model trained on TexTeller 2.0 errors and successively introduce retraining on MML-Net errors, ImageNet-pretrained weights, and the 1024×320 input window. Table 5.2 reports the best test precision and recall obtained for each configuration.

Table 5.2: Progressive ablation of the adaptations introduced for SiaEval-MN.

Configuration	**Test Precision (%)**	**Test Recall (%)**
SiaEval-TT on TexTeller 2.0 predictions	94.30	77.81
SiaEval-TT on MML-Net predictions	75.00	0.28
+ Retraining on MML-Net errors	95.91	19.66 (+19.38 pp)
+ ImageNet-pretrained weights	94.64	29.64 (+9.98 pp)
+ 1024×320 input window	94.04	57.32 (+27.68 pp)

The results show that each adaptation contributes to the verification performance on MML-Net errors. Directly applying SiaEval-TT to MML-Net predictions results in almost no recall, corresponding to the results of Figure 5.9, confirming that the representation learned from TexTeller 2.0 errors does not transfer effectively to the MML-Net error distribution. Fine-tuning the verifier on MML-Net errors restores the ability to distinguish the two classes, increasing recall to 19.66%. Using ImageNet-pretrained weights further increases recall to 29.64%. The largest improvement is obtained by

replacing the square 512^2 input with the 1024×320 window, which increases recall to 57.32% while maintaining 94.04% precision.

These results indicate that the adaptation to MML-Net benefits from both learning from model-specific errors and improving the visual representation. In particular, the larger rectangular input window provides the largest individual gain in recall, suggesting that preserving fine-grained visual details is important for detecting the subtle errors produced by MML-Net.

5.6.4 Discussion on Model-Specific Adaptation

Overall, the experiments highlight two main findings regarding the adaptation of SiaEval to different OCR models. First, the learned similarity representation is not fully model-independent. The results show that SiaEval-TT does not transfer directly to MML-Net, whose errors are fewer but more subtle. The different error distributions require a learned representation specifically adapted to these fine-grained differences. The intuition that SiaEval-TT would be sufficiently general to detect MML-Net errors without retraining is therefore not supported by the results.

Second, the verification framework can nevertheless be adapted to a new OCR model without redesigning the entire verification pipeline. Retraining SiaEval on MML-Net errors substantially improves its ability to detect these more subtle errors, while requiring only a lightweight retraining procedure and limited architectural changes. Only the encoder and input preprocessing are adapted to the target error distribution.

An existing limitation concerns the adaptation of SiaEval to human transcriptions. Available production data do not contain enough labels of actual human transcription errors. Since the manual workflow targets approximately 99.99% accuracy, genuine errors are extremely rare in the available data. No current experiments can therefore establish whether SiaEval could reliably detect errors in manual transcriptions.

5.7 Conclusion and Future Work

This chapter introduced SiaEval, a reference-free post-OCR verification framework for PMER. Instead of comparing an OCR prediction with textual ground-truth, SiaEval compares the input expression image with an image rendered from the predicted MathML and learns a visual similarity metric through a Siamese network.

The experiments on errors from the TexTeller model demonstrate that a lightweight ResNet-18 Siamese network can distinguish correct and incorrect predictions under large variations in font, scale, and image quality. The model achieves a recall of 77.81% at 95.30% true negative rate, outperforming a zero-shot GPT-5-mini baseline.

The experiments with errors from our MML-Net model show that the verifier is

sensitive to the distribution of OCR errors used during training. The model trained on TexTeller errors, SiaEval-TT, does not transfer to MML-Net errors, whose errors are fewer and more subtle. Retraining the verifier on MML-Net errors and adapting the encoder improves performance, demonstrating that SiaEval can be adapted to a new recognition model while preserving the same verification framework.

For future work, larger and more diverse encoders, additional training data, and hard-negative mining could improve the detection of subtle recognition errors. Curriculum learning could also be used to progressively introduce increasingly difficult negative examples. Second, preprocessing could be improved by automatically removing irrelevant elements such as equation numbers and by better handling very large or degraded expressions. Finally, incorporating textual information from the predicted MathML alongside visual features could help distinguish subtle mathematical errors that are difficult to detect from appearance alone.

Chapter 6

Formulasort and Formuladif: Facilitating Manual Transcription of Mathematical Expressions

6.1 Introduction

The previous chapters addressed two complementary aspects of a reliable recognition system: improving the recognition model and estimating the robustness of its predictions in production. However, even a high-performing automatic system cannot fully automate the process when very high quality requirements must be met. Consequently, a subset of mathematical expressions still requires human intervention for verification or correction. In the Luminess production workflow, mathematical expressions are manually reviewed and validated by trained operators through a process that is already functional and provides reliable results (See Section 1.2).

This chapter introduces two contributions developed specifically for the Luminess production pipeline: **Formulasort** and **Formuladif**. They aim to exploit the structure, redundancy, and similarity present in documents to organize and assist human intervention in a way that simultaneously reduces its cost and the risk of error.

The first contribution aims to reduce repetitive manual operations by ordering mathematical expressions according to their visual similarity. By grouping similar expressions, Formulasort facilitates the reuse of previously typed expressions by operators and reduces the cognitive effort associated with rapidly switching between different types of mathematical structures.

The second contribution, Formuladif, generates visual difference images between consecutive expressions. Its purpose is to support operators when almost identical expressions are processed consecutively, particularly when they use copy-paste. A clear image highlighting subtle differences between two successive images helps prevent inattentive copy-pasting and the introduction of additional errors.

Although these contributions address our specific industrial workflow, their underlying methods rely on general computer vision algorithms, including image similarity metrics, hierarchical clustering, and a decision model trained to reproduce human decisions and predict the relevance of automatically generated difference images.

6.2 Motivation for Formulasort and Formuladif

To help us design new assistance tools for mathematical expression transcription that are useful to the production team, we observed the production workflow and interviewed operators working at the Luminess production site in Romania. The objective was to better understand the limitations of the existing human-in-the-loop workflow and find improvement opportunities. Due to confidentiality, the detailed organization of the production process and the complete interview discussions cannot be fully disclosed. However, the main observations relevant to the design of Formulasort and Formuladif are presented in this section.

For each patent, mathematical expressions are first manually segmented and stored according to their original order of appearance in the patent document. These extracted expressions form a sequence of equations that operators progressively transcribe into MathML through the production interface (Figures 6.1 and 6.2 illustrate this process). To limit the length of this task, operators currently process expressions in small batches of 10 formulas. The mathematical expressions are taken in the order in which they appear in the patent document and divided into batches following this order. This prevents operators from having to manually handle very large documents containing thousands of expressions in a single session.

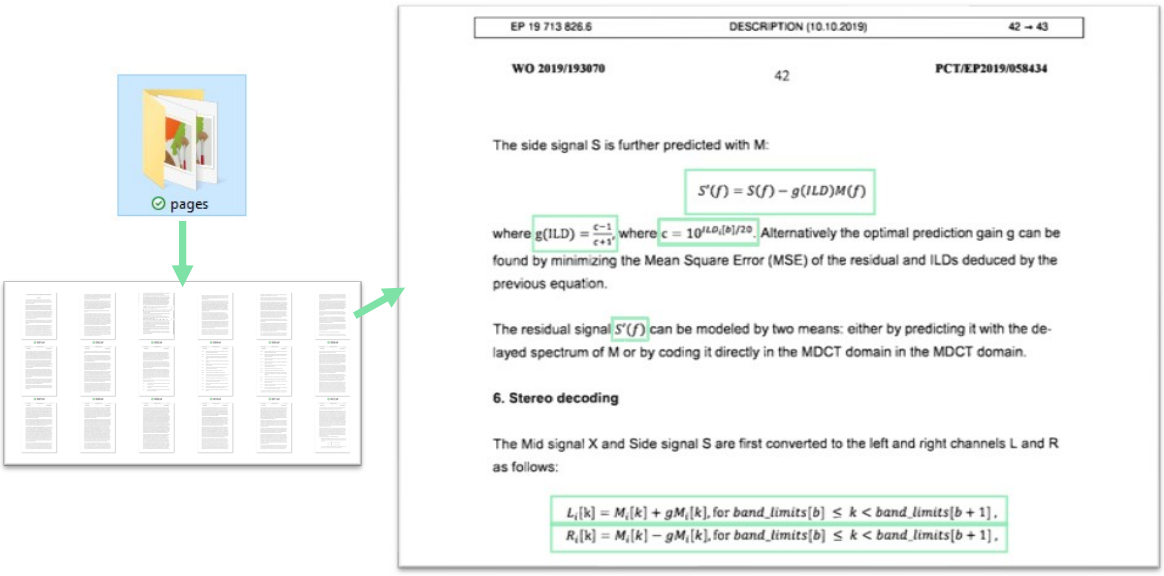


Figure 6.1: An illustration of the segmentation of mathematical expression images from patent applications.

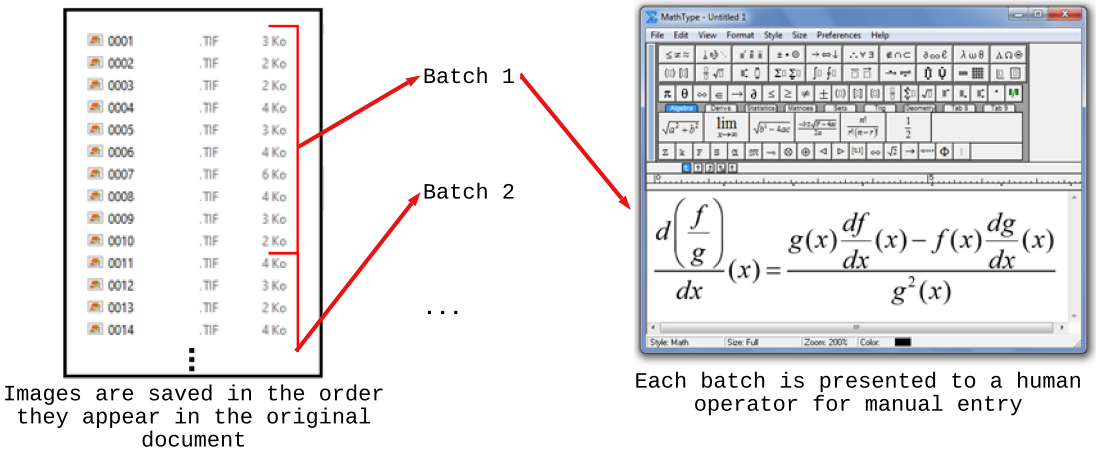


Figure 6.2: Illustration of the batch creation and transcription process.

Many patents contain repeated or nearly identical mathematical expressions. Duplicates are particularly frequent in patent claims, which are published in the three official

EPO languages: French, English, and German. The same mathematical expression may appear several times within the same patent document, once in each language version of the claims.

Expressions with similar mathematical structures often require similar MathML symbols, keyboard shortcuts, and interface operations. Processing such expressions sequentially could allow operators to reuse the same shortcuts and interface elements, reducing the time spent searching for symbols and decreasing the cognitive effort associated with switching between different mathematical structures and menus. Placing identical or highly similar expressions consecutively also facilitates the use of the copy-paste tool. When the same mathematical expression appears multiple times, an operator can directly reuse the transcription of an expression that was typed immediately before, instead of transcribing it from scratch.

To quantify the potential benefits of reusing previously transcribed formulas, an analysis was conducted on the test set of the PatentME-600k dataset presented in Chapter 4, comprising 66,959 mathematical expressions extracted from 23,024 distinct patents. As shown in Table 6.1, 57,163 formulas occur only once, while 3,886 occur multiple times within individual patents (i.e., they have the exact same MathML markup). By transcribing each repeated group only once and reusing the resulting MathML through copy-paste, 5,910 formula occurrences, corresponding to 8.83% of the 66,959 formulas, could be processed through copy-paste instead of being manually transcribed from scratch.

Table 6.1: Distribution of duplicate mathematical expressions within individual patents in the PatentME-OCR dataset.

Occurrences	Expressions
1x	57,163
2x	2,907
3x	599
4x	194
5x	70
6x	45
7x	18
8x	17
9x	9
10x	6
$>$ 10x	21
Total duplicates:	3,886

However, this increased use of copy-paste introduces an additional risk. When two consecutive expressions are similar but not identical, copying an existing transcription and manually modifying it may be more error-prone than entering the expression from

scratch. Subtle differences between two expressions can be unnoticed during a rapid visual comparison, potentially resulting in an incorrect transcription.

Operators were frequently observed rapidly flicking between the previous and current expressions in the production interface to compare them visually and identify differences. This rapid alternation between expressions can help identify small differences but is dependent on human perception and may fail when the modifications are subtle.Quality control also showed a higher error rate for expressions that had been copy-pasted compared with expressions typed from scratch.

All of these observations highlight the need for an additional visual cue that can make subtle differences more explicit.

Sorting by similarity may have a dual effect. It can reduce transcription effort by allowing greater reuse of previous expressions and making the process faster, but it can also increase the risk of transcription errors when similar expressions have subtle differences. This trade-off motivates the two complementary contributions introduced in this chapter:

- **Formulasort**: a similarity-based ordering method designed to group identical and similar mathematical expressions. By placing related expressions consecutively and in the same batch, Formulasort reduces the cognitive effort required to use the input interface and facilitates the use of copy-paste.
- **Formuladif**: a visual comparison method that generates difference images between consecutive expressions. Formuladif provides an explicit visualization of similarities and differences by highlighting modified regions between two expressions. This additional information is particularly important after Formulasort, since grouping similar expressions may increase the use of copy-paste operations and consequently the risk of overlooking subtle differences.

Formulasort exploits similarity to reduce cognitive and interaction costs, whereas Formuladif makes differences explicit to reduce the error risk introduced by this increased reuse. Figure 6.3 illustrates the proposed approach on a patent containing multiple similar mathematical expressions. The first row shows the original order of the expressions, and the second shows their order after applying Formulasort and the difference images generated by Formuladif. This example shows how similar expressions can be grouped together and how the differences between consecutive expressions can be made explicit.

The methodology used to develop and evaluate these two tools, as well as quantitative results on their performance, is described in the following sections.

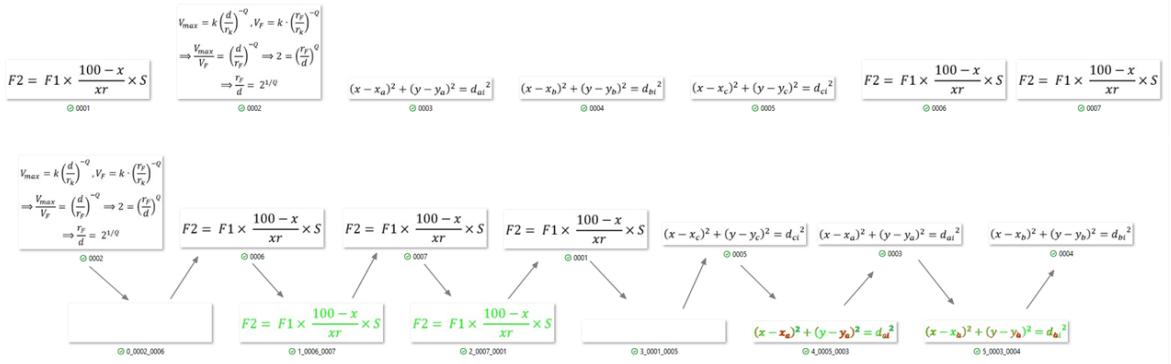


Figure 6.3: Example of the reordering performed by Formulasort. The top row shows the original order, while the bottom row shows the reordered expressions and generated difference images.

6.3 Formulasort

6.3.1 Methodology

Overview

The objective of Formulasort is to automatically determine a transcription-efficient ordering of the mathematical expressions contained in a patent, such that visually similar expressions are presented consecutively.

Let

$$X = (x_1, x_2, \ldots, x_N)$$

denote the sequence of mathematical expressions in a patent application. This order comes from the reading order of the document and is not necessarily optimal for the transcription task. In particular, visually similar or identical expressions may be separated by unrelated expressions, requiring operators to repeatedly switch between different shortcuts on their virtual keyboard and making it less obvious when the copy-paste functionality can be used between expressions.

Formulasort aims to produce a new ordering

$$X' = (x'_1, x'_2, \ldots, x'_N)$$

of the same expressions, such that visually similar expressions are placed closer together. Given a distance function d between expressions, the quality of an ordering can be expressed by the cumulative distance between consecutive expressions:

$$S(X') = \sum_{i=1}^{N-1} d(x'_i, x'_{i+1}).$$

To achieve this, we use agglomerative hierarchical clustering with single linkage, a clustering method for which the distance between two clusters is defined as the minimum distance between any pair of observations belonging to the two clusters [188]. This choice allows expressions with different visual characteristics to be grouped when they are connected through pairs of similar expressions, rather than requiring all expressions within a group to be mutually similar. Agglomerative hierarchical clustering iteratively compares feature vectors to build a clustering dendrogram, as illustrated in Figure 6.4. The leaves of the dendrogram can then be read from left to right to obtain an ordered list of mathematical expressions, with similar expressions placed consecutively.

Formulasort tries to minimize the cumulative distance by generating an ordering from the hierarchical clustering dendrogram. Visual features are extracted from the images of the mathematical expressions and pairwise distances are computed between them. The resulting distances are used for hierarchical clustering, and the leaves of the dendrogram are read from left to right to obtain a new order for presenting the expressions to the operators.

Our remaining task is to identify the best feature extraction method for our mathematical images and the best distance metric to produce relevant orderings. We also need to define a way to quantify the quality of an ordering compared with the original order of the patent to evaluate the different methods.

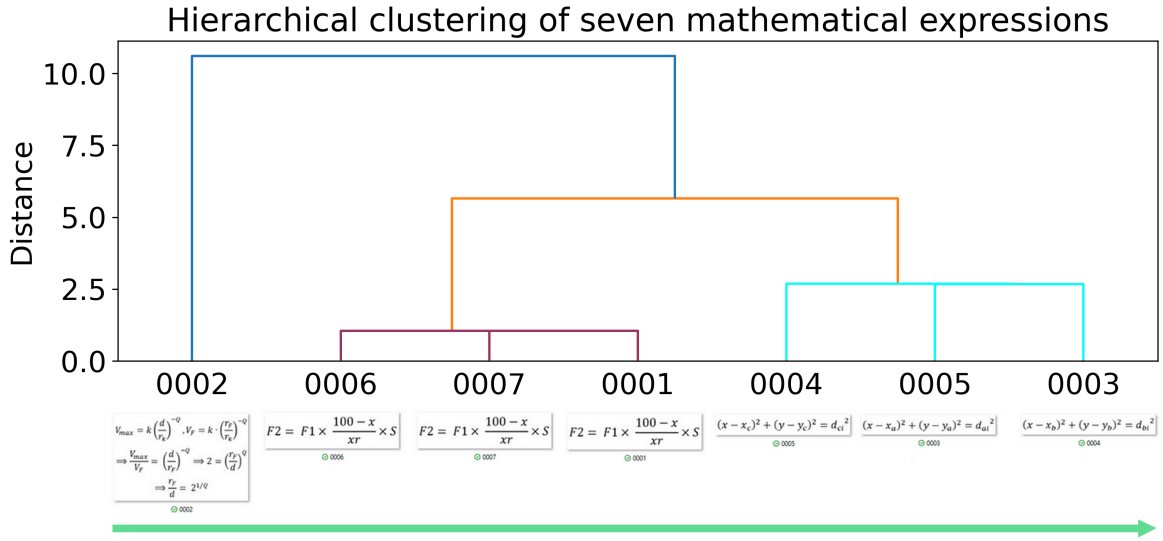


Figure 6.4: A dendrogram resulting from hierarchical clustering of mathematical expression images, comparing extracted visual features. The new order for presenting the expressions to the human operator can be read from left to right following the green arrow.

Formulasort dataset

A dedicated dataset composed of 994 patent documents was created for Formulasort. We randomly selected 80% of the documents for training and 20% for testing. The resulting training set contains 792 unique patents and 9,032 mathematical expressions, while the test set contains 202 unique patents and 2,055 mathematical expressions. The distribution of the number of mathematical expressions per patent is shown in Figure 6.5. A two-sample Kolmogorov–Smirnov test was performed to verify that the two datasets have similar distributions of number of expressions per patent. The test resulted in a KS statistic of $D = 0.0305$ and a p-value of 0.9970, suggesting that the train and test sets have similar distributions in terms of the number of mathematical expressions per patent.

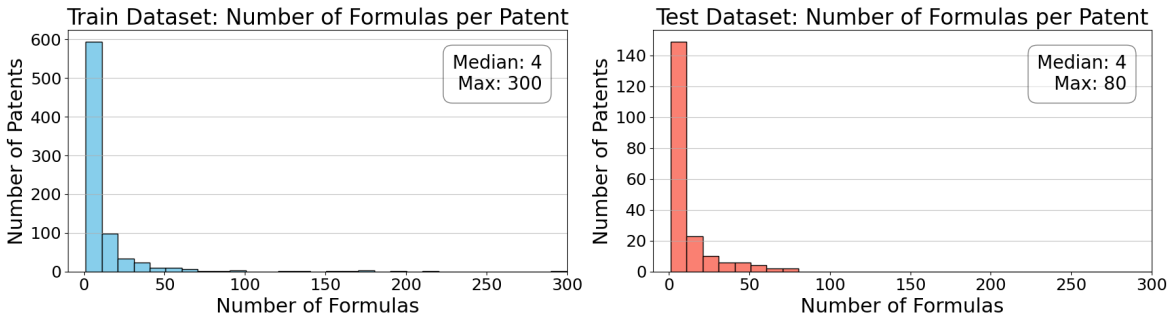


Figure 6.5: Distribution of the number of mathematical expressions per patent in the training and test datasets.

Each patent contains a list of images of mathematical expressions from the patent, stored in the order in which they appear in the text, as well as the MathML markup of each expression. The MathML markup is not available in production, as Formulasort sorts the images before the mathematical expressions are transcribed. It is used as a reference to evaluate the quality of the resulting order. The original order represents the order in which the expressions would be presented to an operator without Formulasort. The training set is used to train the supervised approaches, while the test set is used to evaluate and compare all models.

Image representations

To use hierarchical clustering, it is necessary to define how the mathematical images are represented and compared. Two main approaches are considered: pixel-based image similarity and CNN-based feature representations, with pretrained and fine-tuned variants. Unlike in the SiaEval chapter, the images being compared come from the same document, so there is generally less heterogeneity in terms of font, size, and noise level. However, variations may still be present and must be taken into account when defining

the similarity between two expressions. For the pixel-based approach, the mathematical images are resized to 256 × 256 pixels and flattened into a one-dimensional vector.

For the CNN-based approach, feature vectors are extracted from different CNN backbones. Our approach is similar to the feature comparison employed in [156]. We compare three CNN models: ResNet18 [164], MobileNetV3-Small [189], and SqueezeNet [190]. These architectures were selected to provide different levels of model size and computational cost while remaining suitable for CPU-based inference. In addition, hierarchical clustering has a computational complexity of $O(n^2)$ with respect to the number of expressions. Using lightweight CNN backbones reduces the time required to extract the feature vectors and makes the complete sorting process faster, particularly for patents containing a large number of mathematical expressions. The three architectures also provide a comparison between a relatively larger lightweight model (ResNet18) and smaller architectures (MobileNetV3-Small and SqueezeNet). For each CNN architecture, a fine-tuned version is trained using a supervised metric-learning target based on the MathML associated with each training image. Figure 6.6 illustrates the different visual feature extraction methods considered.

The resulting representations, either flattened images or CNN feature vectors, are compared using cosine or Euclidean distance to evaluate the effectiveness of these distance metrics for Formulasort.

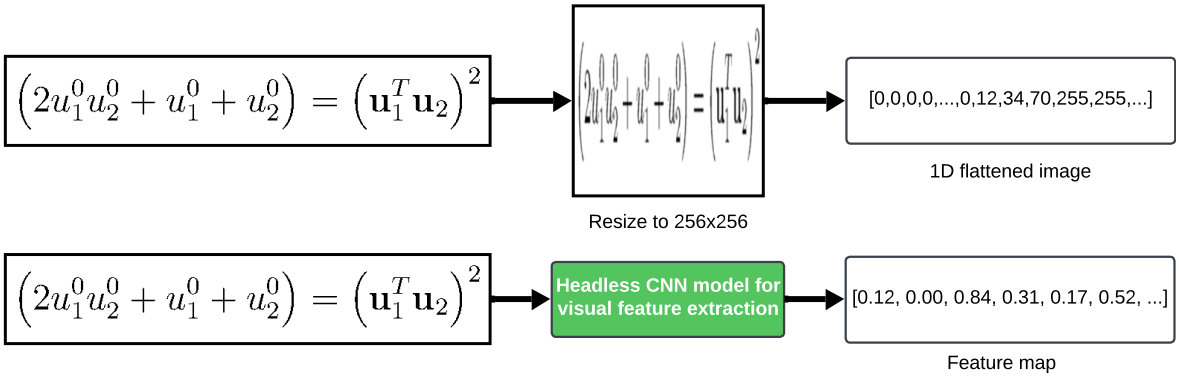


Figure 6.6: Illustration of our different methods for visual feature extraction for Formulasort.

CNN fine-tuning with a proxy task

To fine-tune the CNN models for our sorting task, a proxy target is needed to guide the learning of a more suitable representation of mathematical expressions. Since no ground-truth of expression ordering is available, we use the textual similarity between MathML expressions as a proxy. The Levenshtein distance between two MathML representations gives a measure of their structural difference. This measure does not perfectly reflect visual similarity because of potential polymorphic ambiguity, where

different MathML structures can produce visually similar renderings. Nevertheless, it provides a signal that can be used to guide the CNN towards representations more suitable for grouping related expressions.

To do so, we use triplet loss [191]. For each patent, the raw Levenshtein distance is computed between all pairs of MathML expressions. Training triplets are then constructed by selecting an anchor expression, a positive expression with a small Levenshtein distance to the anchor, and a negative expression with a larger distance. Only triplets for which the difference between the negative and positive Levenshtein distances is at least 10 are retained. This empirical threshold imposes a minimum structural separation between the positive and negative examples, avoiding triplets for which the distinction between the two is too small.

The triplets are constructed independently within each patent, so that the structural relationships used for training are derived from expressions belonging to the same document. The number of mathematical expressions varies across patents, with a median of 4 expressions and up to 300 expressions in the biggest patent. At most 50 expressions are sampled from each patent (see Figure 6.7) to limit the computational cost of pairwise distance computation and triplet generation, and prevents patents with very large numbers of expressions from contributing disproportionately to the training. For most patents, this limit does not affect the available expressions, while it limits the contribution of the largest patents.

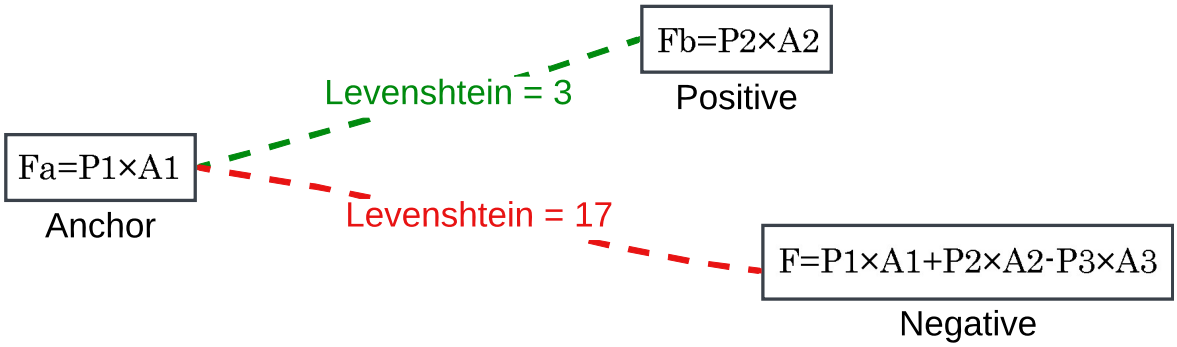


Figure 6.7: Illustration of the triplet generation process. An anchor expression is paired with a positive expression and a negative expression based on their Levenshtein distances. The displayed Levenshtein distances are illustrative.

The CNN is initialized with ImageNet-pretrained weights and fine-tuned with a two-layer projection head. The backbone produces a feature vector whose dimension depends on the architecture. The projection head maps this feature vector to a 128-dimensional embedding, which is L2-normalized and used for metric learning. For each triplet, Euclidean distances are computed between the anchor and the positive and between the anchor and the negative in this normalized embedding space:

$$\mathcal{L}_{\text{triplet}} = \frac{1}{N} \sum_{i=1}^{N} w_i \max\left(0,\, d(E_a^{(i)}, E_+^{(i)}) - d(E_a^{(i)}, E_-^{(i)}) + m\right),$$

where E_a, E_+, and E_- denote the normalized 128-dimensional embeddings of the anchor, positive, and negative expressions, respectively, $d(\cdot, \cdot)$ is the Euclidean distance, and N is the total number of triplets in the batch. The margin is set to $m = 0.2$. During training, the CNN backbone and the projection head are jointly optimized using this loss.

The loss weight w_i is defined as the difference between the raw Levenshtein distances of the negative and positive expressions:

$$w_i = d_{lev}(x_a^{(i)}, x_-^{(i)}) - d_{lev}(x_a^{(i)}, x_+^{(i)}).$$

Here, x_a, x_+, and x_- denote the MathML representations of the anchor, positive, and negative expressions, respectively. We assume that triplets with a larger difference in MathML provide a stronger supervisory signal and assign them a greater weight during training.

To limit the computational cost of triplet generation, 20,000 triplets are randomly sampled at each training epoch, and each CNN is fine-tuned for five epochs using a batch size of 32.

After fine-tuning, the projection head is discarded, and the feature vector produced directly by the CNN backbone is used as the representation for Formulasort. This follows the recommended use of projection heads in representation learning, where the projection head provides a task-specific space for the training objective while the pre-projection representation can be used for the actual tasks [192]. The 128-dimensional embedding is used as a temporary space for metric learning, and the richer backbone feature vector is used for distance computation and hierarchical clustering.

The three supervised representations are named ResNet18-finetuned, MobileNetV3-Small-finetuned, and SqueezeNet-finetuned. They are compared with their corresponding ImageNet-pretrained versions with no fine-tuning.

The extracted feature vectors are used to compute pairwise distances between the mathematical expressions of each patent. These distances are then used as input to agglomerative hierarchical clustering with single linkage, using either Euclidean or cosine distance to compare the resulting orderings. This produces a dendrogram representing the hierarchical similarity relationships between the expressions.

The dendrogram provides a hierarchical structure and not a unique linear ordering. To obtain a linear ordering for Formulasort, the expressions are taken in the order of the leaves returned by the hierarchical clustering algorithm. These leaves are read from left to right in the resulting dendrogram representation, as illustrated in Figure 6.4. This leaf ordering defines a new sequence of mathematical expressions presented to the

operator. In this way, the hierarchical similarity structure provided by the clustering is converted into the linear sequence used by Formulasort.

Quantitative evaluation of the sort quality

We need a method to characterize the resulting ordering. For an ordering containing N expressions, the cumulative distance between consecutive feature vectors is computed as:

$$S = \sum_{i=1}^{N-1} d(E_i, E_{i+1}),$$

where d denotes the considered distance (cosine or Euclidean), and E_i denotes the feature representation of the i-th mathematical expression in the considered ordering. The value of S represents the cumulative distance travelled through the feature space when following the expressions in the given order. It provides a measure of the global coherence of the ordering with respect to the considered visual representation. A lower value indicates that the considered ordering follows a shorter trajectory through the feature space.

Two values are computed: $S_{original}$, the cumulative distance along the original patent order, and S_{sorted}, the cumulative distance along the ordering produced by Formulasort. The cumulative-distance reduction is defined as:

$$\Delta S = \frac{S_{original} - S_{sorted}}{S_{original}}$$

A positive value indicates that Formulasort produces a shorter trajectory through the considered feature space than the original order, while a negative value indicates a longer trajectory. This metric is computed for each feature representation and distance–metric pair.

However, the visual distance used for sorting cannot be used alone to determine whether the resulting order is correct, as it only provides information about visual similarity. In principle, if hierarchical clustering performs well, the cumulative visual distance should mostly be reduced after sorting. To provide a proper ground-truth evaluation metric, we use the MathML markups. Since the MathML associated with each expression is available in the evaluation dataset, the Levenshtein distance between the MathML markups of consecutive expressions can be computed for both the original and sorted orders. The cumulative MathML distance is defined as:

$$S_{lev} = \sum_{i=1}^{N-1} d_{lev}(x_i, x_{i+1})$$

where d_{lev} is the Levenshtein distance between two consecutive MathML representa-

tions x_i and x_{i+1}. We use the unnormalized Levenshtein distance so that the absolute amount of change between two expressions contributes directly to the score. Longer expressions can contribute more to the cumulative distance than shorter ones. This choice ensures that a larger number of structural changes in a long expression is not assigned the same weight as a smaller number of changes in a short expression. The cumulative Levenshtein distance reflects the total amount of structural change between consecutive expressions and is used here as a proxy for the cognitive load associated with switching between expression types during transcription.

The same improvement measure is then computed:

$$\Delta S_{lev} = \frac{S_{lev,original} - S_{lev,sorted}}{S_{lev,original}}$$

A positive ΔS_{lev} indicates a lower cumulative structural distance between consecutive mathematical expressions according to their MathML representations. It indicates an improvement in the structural continuity of the browsing order. When $\Delta S_{lev} = 0$, the original and sorted orders have the same cumulative distance and are considered equivalent. In this case, the original order is retained.

For each patent, both the visual improvement ΔS and the MathML improvement ΔS_{lev} are computed. To evaluate whether a sorting method improves the browsing order, each improvement is converted into a binary outcome based on its sign: a positive value indicates an improvement, while a negative value indicates a degradation. This binary evaluation makes it possible to report the proportion of patents for which a method improves the ordering.

This binary criterion is a slight limitation of our evaluation protocol, as any positive improvement is considered equally successful regardless of its magnitude. For example, a very small positive ΔS_{lev} and a large positive ΔS_{lev} are both counted as improvements. The magnitude of the improvement is nevertheless saved in the computed scores and could be analyzed separately.

The proportion of patents for which both the visual score and the MathML score indicate an improvement, i.e., $\Delta S > 0$ and $\Delta S_{lev} > 0$, is reported. This value represents the proportion of cases in which Formulasort generates an ordering that is improved according to both the visual metric and the MathML ground-truth. A patent may also have a negative visual ΔS while having a positive MathML ΔS_{lev}. In this case, the visual metric indicates that the sorting is not beneficial, even though the resulting order is actually better according to the MathML ground-truth. Conversely, a positive visual ΔS combined with a negative MathML ΔS_{lev} represents a case where the visual metric incorrectly indicates an improvement. By considering the MathML criterion as the reference criterion, we report confusion matrices and the resulting precision, recall and F1 scores. They provide a direct measure of how reliably the visual evaluation identifies

orders that are actually improved. In production, only the visual criterion ΔS can be used to accept or reject a proposed new order. It is important to evaluate its reliability and its alignment with the textual MathML criterion ΔS_{lev}.

6.3.2 Results of the different feature extraction methods

Table 6.2 reports the F1 scores obtained on the test set for each feature extraction method and distance metric used for hierarchical clustering. These results show that the fine-tuned ResNet18 using the Euclidean distance achieves the highest F1-score of 84.95% for predicting MathML-based improvements from the visual criterion.

Table 6.2: F1-score obtained on the independent test set for each feature extraction method and distance metric. The best value for each row is shown in bold. Models are ranked by their best F1-score across the two distance metrics.

Feature configuration	**Cosine**	**Euclidean**
ResNet18-finetuned	84.62%	**84.95%**
image_256_flat	79.31%	**82.49%**
squeezenet-finetuned	81.11%	**82.49%**
SqueezeNet	**80.85%**	77.19%
mobilenet_v3_small-finetuned	79.12%	**79.35%**
ResNet18	**79.10%**	77.91%
MobileNetV3-Small	**78.41%**	74.56%

For the pretrained CNN models, cosine distance consistently provides better performance than Euclidean distance. SqueezeNet, ResNet18, and MobileNetV3-Small reach F1-scores of 80.85%, 79.10%, and 78.41% with cosine distance, compared with 77.19%, 77.91%, and 74.56% with Euclidean distance, respectively. This indicates that cosine distance is more suitable for comparing these pretrained feature representations.

The fine-tuned models achieve their best performance with Euclidean distance. The fine-tuned ResNet18 evaluated with Euclidean distance reaches the highest F1-score of 84.95%. This suggests that fine-tuning changes the learned representations to make Euclidean distance more effective for predicting MathML-based improvements from the visual criterion.

Euclidean distance provides the best results for the simple flattened image representation. The absolute differences between pixel values are better captured by the Euclidean distance. The fine-tuned CNNs also generally perform better with Euclidean distance. This behavior is expected from the training procedure, since the triplet loss used to fine-tune the embeddings relies on Euclidean distances to bring similar expressions closer and separate dissimilar expressions in the embedding space. The resulting representation is optimized so that Euclidean distances reflect the similarity relationships.

The comparison also shows that fine-tuning does not provide the same gain for all architectures. With the Euclidean distance, fine-tuning increases the F1-score by 7.04 pp for ResNet18, more than for SqueezeNet and MobileNetV3-Small (+5.30 pp and +4.79 pp respectively).

In the following experiments, each feature representation is evaluated using the distance metric that achieved the best F1-score for that representation, as reported in bold in Table 6.2. In addition, patents containing only one or two mathematical expressions are excluded from the following evaluation, as reordering such a small number of expressions does not affect the operators' work. This reduces the test set from 202 to 123 patents.

Figure 6.8 reports the proportion of test patents for which Formulasort improves or degrades the ordering according to ΔS_{lev}. The fine-tuned ResNet18 achieves the highest proportion of improved orderings, with 84 out of 123 patents (68.9%), followed by the simple image size baseline (67.2%).Overall, all feature representations improve the ordering for around two thirds of the test set, while the fine-tuned ResNet18 provides the most improvements.

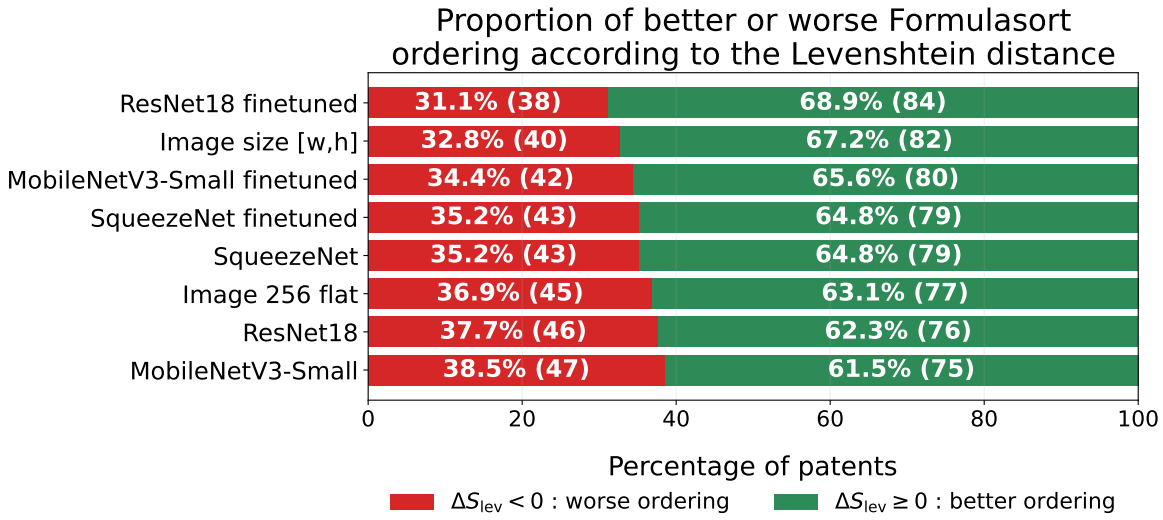


Figure 6.8: Proportion and number of test-set patents for which Formulasort improves or degrades the ordering according to ΔS_{lev}.

Figure 6.9 provides a more detailed view of the ordering results by comparing the sign of the sorting improvement measured in the visual feature space, ΔS, with the sign of the improvement measured using the cumulative MathML Levenshtein distance, ΔS_{lev}. These confusion matrices show whether orderings considered better according to the visual feature representation are also better according to the Levenshtein-based score, which is used as the ground-truth for evaluating the sorting quality.

The true positive (top-right) and true negative (bottom-left) cases correspond to patents for which the visual criterion and the MathML criterion agree on whether

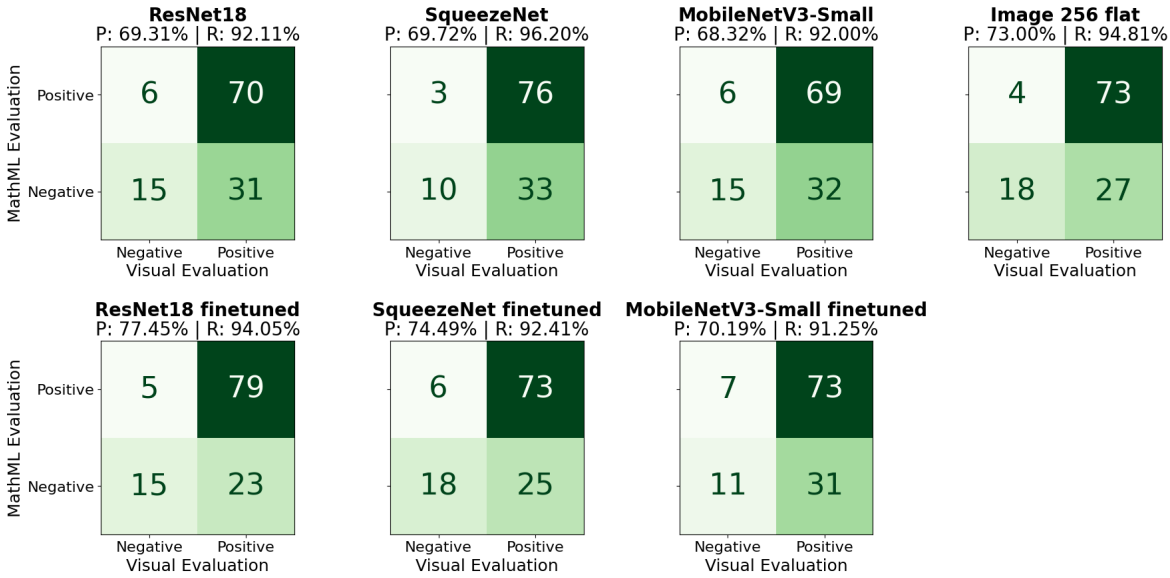


Figure 6.9: Confusion matrices obtained on the independent test set for each model. Each cell of the matrix indicates if the resulting ordering improves ($\Delta S > 0$) or degrades ($\Delta S < 0$) the cumulative distance between consecutive expressions, evaluated according to the visual feature used for sorting (x-axis) and the MathML Levenshtein distance used as ground-truth (y-axis). Values correspond to the number of patents. P and R represent the precision and recall.

the ordering improves or degrades the image sequence. False positive (bottom-right) and false negative (top-left) cases correspond to disagreements between the visual representation and the MathML ground-truth. These disagreement cases are particularly informative because MathML is not available during the actual sorting process. The ability of a visual representation to correctly identify orderings that also improve the MathML-based criterion indicates how well it acts as a proxy for mathematical similarity and whether it can be trusted in production.

ResNet18-finetuned obtains the highest number of true positives, with 79 patents correctly identified as improved according to both the visual criterion and the MathML ground-truth. The comparison between pretrained and fine-tuned model matrices shows fine-tuning generally increases the number of true positives while slightly reducing the number of false positives. For SqueezeNet however, fine-tuning leads to a small decrease in the number of true positives.

6.3.3 Discussion

The results demonstrate that mathematical expressions can be effectively reordered using visual information alone, without requiring access to their MathML representations during the sorting process. The best configuration, based on the fine-tuned ResNet18 representation and Euclidean distance, reaches an F1-score of 84.95% on the independent

test set. This learned visual representation captures the information required to produce an ordering consistent with the mathematical similarity measured from MathML. Fine-tuning using a proxy metric improved the performance of all three evaluated CNN architectures, although with different magnitudes in the improvement.

Formulasort provides a relatively lightweight method for generating sufficiently good sorts of mathematical expressions to simplify manual transcription by operators. However, the computational cost also needs to be considered. The current evaluation focuses primarily on ordering quality, while CNN feature extraction and hierarchical clustering can become computationally expensive when processing large numbers of expressions. The actual processing time should be measured on the hardware used in the production environment.

To implement Formulasort on the actual pipeline we need to take into account patents containing a very large number of mathematical expressions. Hierarchical clustering has a computational complexity of $O(n^2)$ in the number of expressions. Some patents contain several hundreds or even thousands of mathematical expressions, making clustering computationally expensive. To improve scalability in production, expressions are first sorted according to their dimensions. Expressions are then divided into batches containing at most $N_{\max} = 200$ expressions. Hierarchical clustering is independently applied to each batch. This preprocessing step drastically reduces the computational cost for these rare cases while preserving the possibility of grouping visually similar expressions within each batch.

Overall, these results show that Formulasort is effective at improving the ordering of mathematical expressions without relying on their MathML representations.

6.4 Formuladif

6.4.1 Methodology

Overview

The objective of Formuladif is to generate visual *difference images* between consecutive mathematical expressions in the order produced by Formulasort. These images are intended to help operators identify small differences between similar expressions, particularly when the transcription of a previous large expression can be reused through copy-paste. In Figure 6.10, the operator reuses the transcription of the previous mathematical expression because it is very similar to the expression that needs to be transcribed, while being sufficiently long that transcribing it from scratch would be time-consuming. In this case, the difference image shown in the upper right is particularly relevant, as it allows the operator to identify at a glance the four subtle

differences that need to be modified for the transcription to be correct, as the two expressions are not strictly identical.

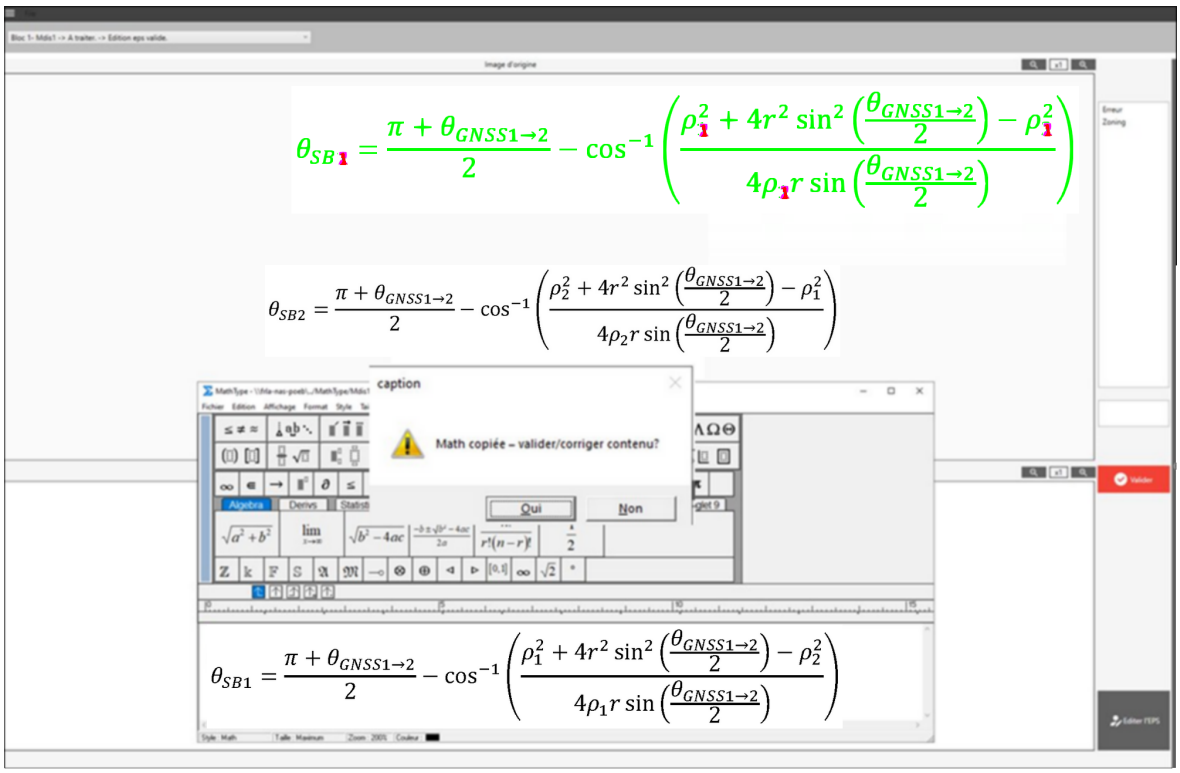


Figure 6.10: Example of a relevant difference image generated by Formuladif. The operator can reuse the transcription of the previous expression and identify the four differences that need to be modified instead of transcribing the expression from scratch or risking missing a difference and publishing an error.

Generation of the difference images

For each pair of successive expression images A and B, two difference images are generated. The first one directly superposes the two images on a blank canvas with the size of image B, and assigns a color to each pixel according to its presence in the two images: white if the pixel is white in both images, green if it is black in both images, red if it comes only from image A, and magenta if it comes only from image B.

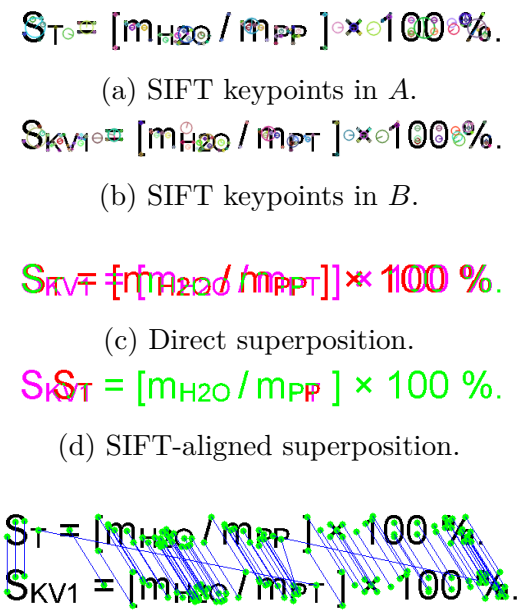
Direct superposition can be affected by small misalignments between the two expressions. Both images may contain very similar content while being shifted horizontally or vertically, causing a direct pixel-to-pixel comparison to produce large red and magenta areas that do not correspond to actual differences in the mathematical expressions.

To reduce these irrelevant differences, a second image is generated after estimating a translation between images A and B using SIFT matching. SIFT detects local keypoints [193] and computes descriptors for both images, which are matched using a FLANN-based matcher [194] and filtered with Lowe's ratio test [193]. The coordinates

of the remaining matches are then used to estimate a linear translation, which is applied to align the two images before generating the second difference image.

Only translation is considered, without rotation or scaling. This restriction is motivated by the fact that the mathematical expressions originate from the same patent documents and are typically produced by the same author using the same software. They can be assumed to share the same orientation and scale. Limiting the transformation to translation avoids introducing unnecessary degrees of freedom and restricts the alignment to the expected type of displacement between the two images.

The generation of these two difference images is illustrated in Figure 6.11. The resulting images simultaneously represent the common content and the differences between the two mathematical expressions. In the Formuladif pipeline, we generate both the difference image from the direct superposition of the two images, and generated with the SIFT superposition. The following selection method allows us to indentify the most visually helping images.



(a) SIFT keypoints in A.

(b) SIFT keypoints in B.

(c) Direct superposition.

(d) SIFT-aligned superposition.

(e) SIFT matches between A and B.

Figure 6.11: Generation of the difference images for two successive mathematical expressions. SIFT keypoints are detected in images A and B and used to estimate a translation between the two images. Direct superposition produces an unusable difference image. After alignment, the resulting difference image provides a more accurate representation of the actual differences between the two expressions.

Difference image filtering

Generating a difference image does not guarantee that the resulting visualization is useful to an operator. Some difference images may contain large amounts of red or magenta pixels caused by image displacement, segmentation differences, or simply large

differences between the two expressions. Such images may be overly colored and difficult to interpret, while providing little useful information to the operator, as illustrated in Figure 6.12. An automatic filtering step is required to remove uninformative difference images and retain those that are relevant for operator verification.

The filtering must determine what is considered relevant and useful in the images and distinguish a legitimate geometric alignment from an attempt to align two genuinely different contents.

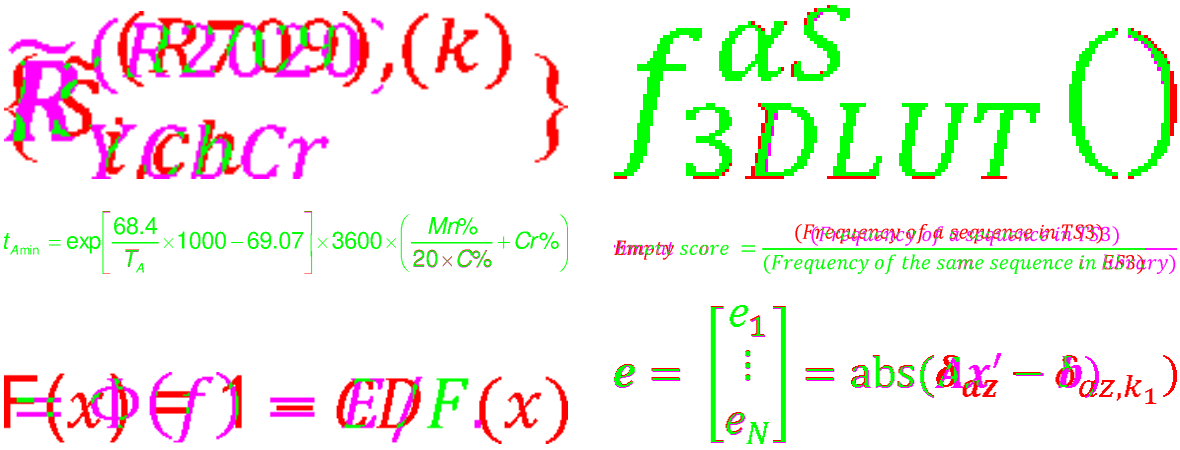


Figure 6.12: Examples of difference images with different levels of usefulness for operator verification. Some images have meaningful differences between expressions, and some others contain irrelevant differences or are difficult to interpret.

To evaluate the usefulness of the generated difference images, we gathered a set of 1000 difference images. They were randomly selected from the patent corpus, with an equal proportion generated using the two difference-generation modes: 500 images obtained by direct superposition and 500 images obtained after SIFT-based alignment, to ensure that both approaches are represented equally in the evaluation of the filtering step.

To evaluate the quality of the generated difference images, a human reference annotation was established by asking an expert operator to annotate each image according to its usefulness for the intended verification task. The operator was provided with an explanation of the Formuladif pipeline and the purpose of the generated difference images before starting the annotation. For each difference image, the operator indicated whether the visualization would be useful or not during the verification process using a binary label, OK or KO.

A dedicated annotation interface was developed to allow the operator to review the generated difference images and record the corresponding relevance decision, as shown in Figure 6.13. Since the annotation was produced by a single expert operator, it should not be considered an objective ground-truth. The labels reflect the operator's opinion and may consequently contain some degree of subjective bias. Nevertheless, this annotation is considered a relevant reference for the evaluation, given the operator's

expertise and familiarity with the verification workflow. These annotations are used as reference labels for evaluating the following filtering models and baselines.

The 1,000 annotated difference images were divided into training and testing subsets, containing 800 and 200 images respectively, while ensuring that all difference images originating from the same patent were assigned to the same subset. The same class distribution was maintained in both subsets, with 60% of the images labeled as relevant by the expert annotator and 40% as irrelevant.

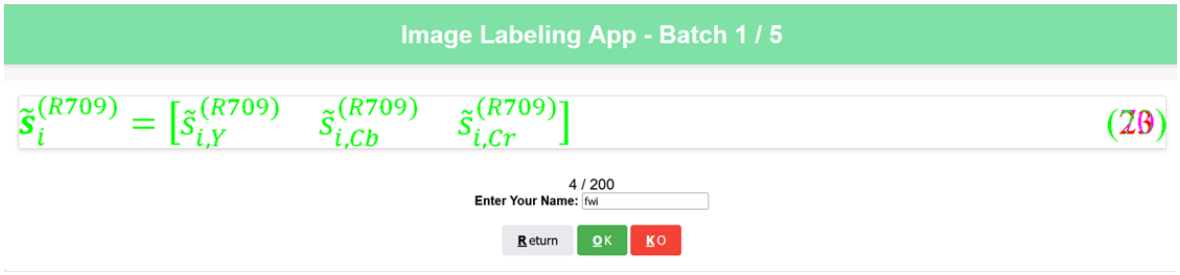


Figure 6.13: Annotation interface used by the expert operator to assess the usefulness of difference images for operator verification.

We define a simple filtering approach based on the proportion of green pixels in the difference image. We compute G_{ratio} as the proportion of green pixels among all colored pixels in the difference image, excluding the white background:

$$G_{\text{ratio}} = \frac{N_{\text{green}}}{N_{\text{green}} + N_{\text{magenta}} + N_{\text{red}}}.$$

For two identical images, all colored pixels are green and $G_{\text{ratio}} = 1$.

A simple baseline filter is obtained by thresholding G_{ratio} at 50%. This threshold provides a simple reference against which the learned filtering models can be compared: if less than half of the colored pixels are green, the difference image is considered too noisy and is rejected. If at least half of the colored pixels are green, the difference image is kept.

However, this simple rule cannot capture all situations in which a difference image may or may not be useful. A difference image containing a large number of red or magenta pixels may still clearly highlight a meaningful modification, while another image with a similar proportion of red and magenta pixels may be visually confusing. The usefulness of a difference image can depend on finer visual characteristics. That is why we trained and evaluated several classification models to predict the relevance of the generated difference images based on the decisions made by the human annotator.

For each difference image, a set of simple image-based features is computed. They are grouped into three categories: color, geometric, and connected-component features. The three groups are defined as follows:

- **Color feature**: *G_ratio* describes the proportion of green pixels among all

colored pixels in the difference image, excluding the white background. This is a simple visual characteristic that is likely used by the human operator when validating the difference images.

- **Geometric features**: These features describe the overall size and shape of the difference image. They consist of the image surface, width, and height, denoted as $img\_surface$, img_width, and img_height. We also include $SIFT\_aligned$, a boolean metadata indicator (True or False) specifying whether the difference image was generated after applying the SIFT-based translation described in Section 6.4.1, or by direct superposition.

- **Connected-component features**: These features describe the size and number of colored connected components in the image. For each color, the average connected-component size and the number of connected components are computed. These features include $avg\_red\_cc\_size$, $avg\_magenta\_cc\_size$, $avg\_green\_cc\_size$, n_red_cc, $n\_magenta\_cc$, and n_green_cc. In particular, many small red or magenta components may indicate scanning noise rather than meaningful differences between images. This information may help the classifier distinguish between noise and actual differences.

The models are evaluated by first using only the color feature G_ratio, then adding the geometric features, and finally the connected-component features. This allows us to measure if adding more features improves the classification.

Three classification models are compared: Logistic Regression, Random Forest, and XGBoost. The models are trained using the 800 images of the train set and evaluated on the 200 images of the test set. The proportion of OK and KO images is preserved in both subsets. Logistic Regression requires feature normalization, so we use a standard scaler in its training pipeline. Random Forest and XGBoost are trained directly on the extracted features.

Hyperparameters are selected using a grid search combined with 5-fold cross-validation on the training set. For each hyperparameter configuration, the training set is divided into five folds for cross-validation. The model is trained on four folds and evaluated on the remaining fold. This process is repeated five times so that each sample is used once for validation. The mean and standard deviation of the precision and F1-score are then computed across the five folds. The tested hyperparameter grids and feature configurations are presented in Table 6.3.

The F1-score is used as the main criterion for hyperparameter selection, as it provides a balance between precision and recall and prevents the classifier from becoming overly conservative and predicting the positive class only for the easiest cases. Once the best hyperparameters have been selected, the corresponding classifier is trained on the

Table 6.3: Feature configurations and hyperparameter grids used for difference image filtering.

Model	**Hyperparameter grid**	**Feature selection**
Logistic Regression	$C \in \{0.01, 0.1, 1, 10, 100\}$ class weight $\in \{\text{None}, \text{balanced}\}$	*G_ratio*, *G_ratio* + geometry, all features
Random Forest	$n_{\text{estimators}} \in \{100, 200, 300\}$ max depth $\in \{\text{None}, 5, 10\}$ min samples leaf $\in \{1, 5\}$ class weight $\in \{\text{None}, \text{balanced}\}$	*G_ratio*, *G_ratio* + geometry, all features
XGBoost	$n_{\text{estimators}} \in \{50, 100, 200\}$ max depth $\in \{1, 2, 3\}$ learning rate $\in \{0.05, 0.1\}$	*G_ratio*, *G_ratio* + geometry, all features

complete training set of 800 images and evaluated on the independent test set of 200 images using accuracy, precision, recall, and F1-score.

For the final test evaluation, precision is considered the primary operational metric, as false positives should be minimized to avoid presenting operators with too many irrelevant difference images. Recall is also reported to ensure that a high precision is not achieved by rejecting too many relevant difference images.

6.4.2 Results of Image Filtering

Quantitative Results

Table 6.4 reports, for each feature configuration, the best hyperparameters selected for each classifier.

Table 6.4: Best hyperparameters for each classifier and feature configuration.

Feature set	Model	Best hyperparameters
G_ratio	Logistic Regression	$C = 0.01$, balanced
G_ratio	Random Forest	100 trees, depth 5, leaf=5, balanced
G_ratio	XGBoost	50 trees, depth 1, lr=0.05
G_ratio + geometry	Logistic Regression	$C = 0.01$, balanced
G_ratio + geometry	Random Forest	200 trees, depth 5, leaf=1, balanced
G_ratio + geometry	XGBoost	100 trees, depth 3, lr=0.05
All features	Logistic Regression	$C = 10$, balanced
All features	Random Forest	200 trees, depth 5, leaf=5, balanced
All features	XGBoost	50 trees, depth 3, lr=0.05

Table 6.5 shows the test results of the classifier–feature combinations. The results should be interpreted as the ability of the filtering methods to reproduce the relevance decisions made by the expert operator. The confusion matrices comparing the operator labels with the predictions of the 50% threshold baseline and the XGBoost model using all features are also reported in Figure 6.14.

Table 6.5: Test performance of classifier–feature configurations.

Feature set	Model	Accuracy	Precision	Recall	F1-score
G_ratio	Thresholding	93.0%	**100.0%**	88.3%	93.8%
G_ratio	Logistic Regression	95.5%	**100.0%**	92.5%	96.1%
G_ratio	Random Forest	95.5%	**100.0%**	92.5%	96.1%
G_ratio	XGBoost	**96.0%**	99.1%	94.2%	**96.6%**
G_ratio + geometry	Logistic Regression	93.0%	95.7%	92.5%	94.1%
G_ratio + geometry	Random Forest	93.5%	95.0%	94.2%	94.6%
G_ratio + geometry	XGBoost	94.0%	95.0%	95.0%	95.0%
All features	Logistic Regression	95.0%	95.8%	**95.8%**	95.8%
All features	Random Forest	95.5%	97.4%	95.0%	96.2%
All features	XGBoost	95.0%	96.6%	**95.8%**	95.8%

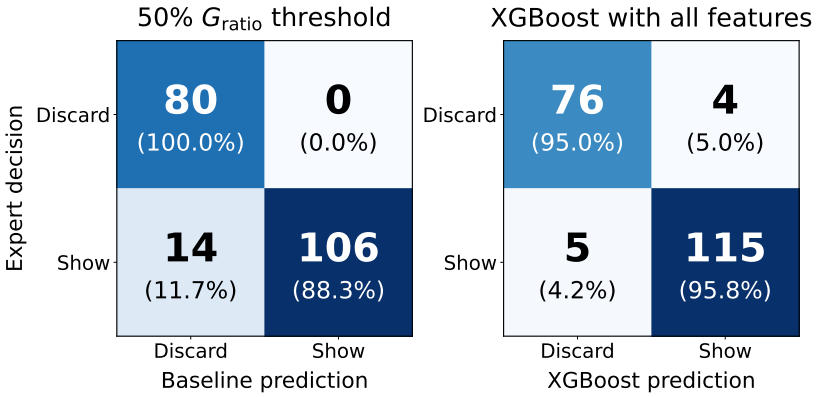


Figure 6.14: Confusion matrices of the 50% G_{ratio} threshold baseline and the XGBoost filtering model using all features, compared with the expert operator's relevance decisions.

The results for the G_{ratio}-only thresholding baseline already reproduces the expert decisions reasonably well, but remains conservative. With 120 relevant and 80 irrelevant images in the test set, the baseline correctly identifies 106 relevant images and rejects all 80 irrelevant images according to its confusion matrix. Its precision is 100%, but 14 relevant images are incorrectly discarded, limiting its recall to 88.3%. The learned models using only G_{ratio} improve over the thresholding baseline. In particular, XGBoost achieves the best performance, with an accuracy of 96.0%, a precision of 99.1%, a recall of 94.2%, and an F1-score of 96.6%.

Adding geometric and other local features does not always improve the F1-score compared with using G_{ratio} alone. For example, the XGBoost model using all features reaches an F1-score of 95.8%, which is lower than the 96.6% obtained with G_{ratio} alone. However, the additional features can improve recall for some classifiers. Logistic Regression and XGBoost both reach 95.8% recall with all features, compared with 92.5% and 94.2%, respectively, when using only G_{ratio}. The confusion matrix of the XGBoost model using all features in Figure 6.14 illustrates this trade-off. Compared with the G_{ratio}-only XGBoost model, it correctly kept more relevant images, increasing

recall from 94.2% to 95.8%, but also kept more irrelevant images, reducing precision from 99.1% to 96.6%. Given the relatively small test set, these differences should be interpreted with caution.

The additional geometric and colour-based features may be useful when the global proportion of green pixels is insufficient to characterize the difference image. For example, an image may contain numerous small red and magenta connected components caused by local image noise while remaining globally readable. Finer-grained features can provide additional information about the structure and distribution of these components. However, the present results do not show that this additional information consistently improves the classification performance over the simpler G_{ratio} representation.

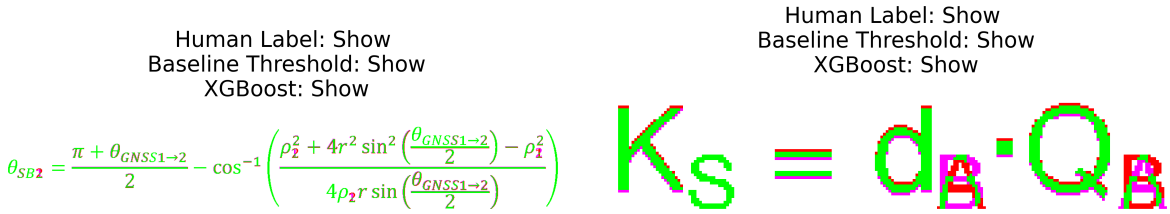
Overall, these experiments show that the filtering models can reproduce the relevance decisions of the expert operator with high accuracy. The XGBoost model using G_{ratio} provides the best balance between retaining relevant images and discarding irrelevant ones, while the models using additional features can achieve higher recall at the cost of lower precision. These results demonstrate that the proposed filtering approach can approximate the expert's selection of useful difference images.

Qualitative Results

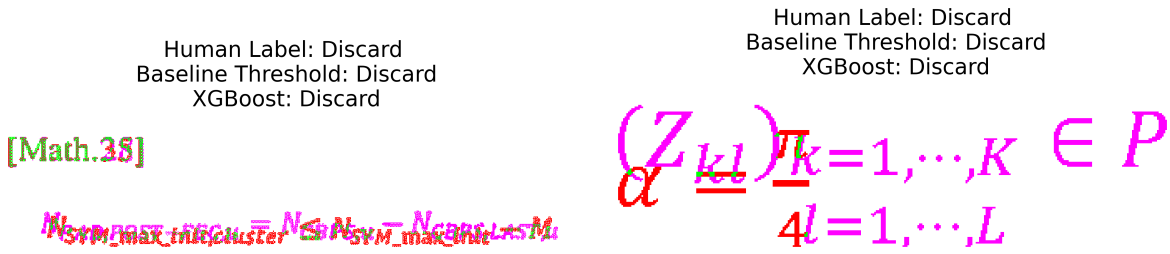
Figure 6.15 illustrates several examples of the decisions made by the human annotator, the 50% threshold baseline, and XGBoost. The first row shows cases where the differences between the two formulas are subtle but clearly visible in the difference image. All three approaches agree that the images should be kept, showing that these difference images can highlight small changes that are relevant for the operator.

The second row shows difference images that should not be displayed, as they provide no useful information to the operator. They are almost entirely composed of red and magenta regions, indicating either that the two compared mathematical expressions have little or no overlapping visual content, or that the images cannot be sufficiently aligned to produce a meaningful comparison. In such cases, displaying the difference image would not help identify a specific change or facilitate the copy-pasting of mathematical content. All three approaches agree that these images should be discarded.

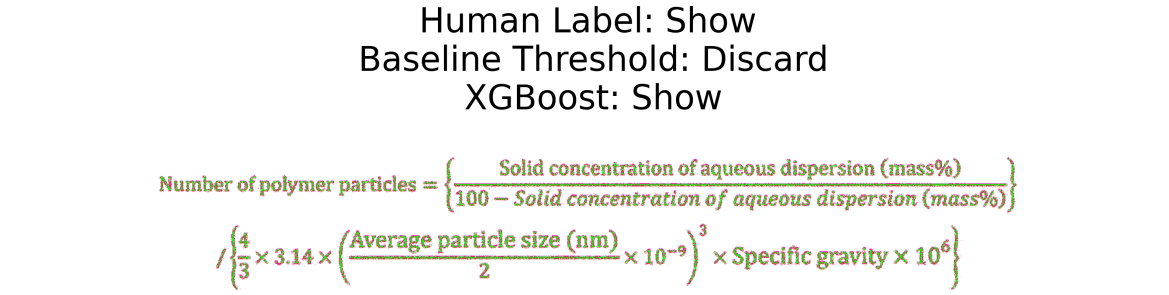
In the third row, the underlying mathematical expressions share many common elements, but scanning noise and bad image quality generate many red and magenta regions. XGBoost agrees with the human annotation and keeps this image, as it can still be useful when copying and pasting mathematical content. The 50% threshold rejects it because the proportion of red and magenta pixels is too high.



(a) **Subtle but meaningful differences.**



(b) **Substantially different expressions.**



(c) **Similar expressions affected by scanning noise.**

Figure 6.15: Examples of difference images and the corresponding filtering decisions.

6.4.3 Discussion and Conclusion

The results obtained by our different approaches should be interpreted with caution. The manually annotated dataset is small, and the ground-truth relies on the judgement of a single expert human annotator. The measured performance can vary depending on the selected examples and human interpretation of ambiguous difference images. A larger annotated dataset would be preferable to better estimate model performance. The main difficulty is to find a way to automatically generate a large quantity of reliable reference annotations for this subjective filtering task. Using the Levenshtein distance between the two underlying MathML representations could provide useful information about the mathematical similarity of the two expressions and could be used to help refine the training of the difference image filtering models. A future direction could be to investigate more powerful models, such as a CNN or ViT,to make better decisions directly from the difference images. Such an approach was not considered here because it would be relatively heavy and because the amount of available training data is currently insufficient.

XGBoost using all features is the model we will use for the filtering stage of the pipeline. Although it does not achieve the highest F1-score, using the complete set of features is expected to produce decisions that are better aligned with human judgement. The additional features allow the model to consider different characteristics of the difference image and make more nuanced decisions than a model using only the amount of green pixels.

This is particularly relevant for ambiguous cases, such as the one in the third row of Figure 6.15, where the difference image contains a large amount of red and magenta noise but can still be useful to the operator. The model learns to imitate the decisions of human annotators rather than relying on a fixed threshold on a single feature.

The visual quality of the generated difference images can be improved through post-processing. The image on the third row in Figure 6.15 could be improved using morphological operations such as erosion, dilation, blurring, or connected-component filtering. Such processing could remove isolated noise and make the visual interpretation easier for the operator.

Formuladif combines automatic difference generation with a learned relevance filtering stage that predicts whether a generated difference image is considered sufficiently informative for human verification. It can also identify difference images that are entirely green, indicating that the two input formulas are identical at the pixel level. This information can be exploited further in the processing pipeline. When identical consecutive expressions are identified, the system could automatically copy the MathML content of the preceding expression to the following expression and only ask the operator to validate the transcription.

Chapter 7

A Semi-Automatic Pipeline for Mathematical Expression Recognition

This chapter uses the contributions presented in this thesis to propose a new semi-automatic processing pipeline for MER in patent applications. We show how the proposed methods can be combined with the existing production workflow at Luminess, and estimate their potential impact on processing time, output quality, and the effort required from human operators.

One additional contribution specific to this new processing chain is the automatic scheduler, presented in Section 7.1. The scheduler estimates whether a mathematical expression is likely to be correctly recognized by the MML-Net OCR model. Expressions predicted to be too difficult for the model are routed to the existing manual transcription workflow, while easier expressions are processed automatically by MML-Net. In Section 7.2, we use the results of all the previous chapters and the PatentME-OCR dataset to simulate patents going through the new production chain and to estimate the improvements provided by the proposed pipeline. Our contributions are deployed so that automation is increased without compromising the very high quality requirements of patent production.

7.1 Automatic Scheduler

7.1.1 Motivation and Problem Formulation

This section introduces a lightweight scheduler designed to estimate, from the image of a mathematical expression alone, whether the MML-Net OCR model described in Chapter 4 is likely to produce a correct transcription. Figure 7.1 illustrates this routing principle: expressions predicted to be easier are processed by MML-Net, while expressions predicted to be more difficult are routed to human operators.

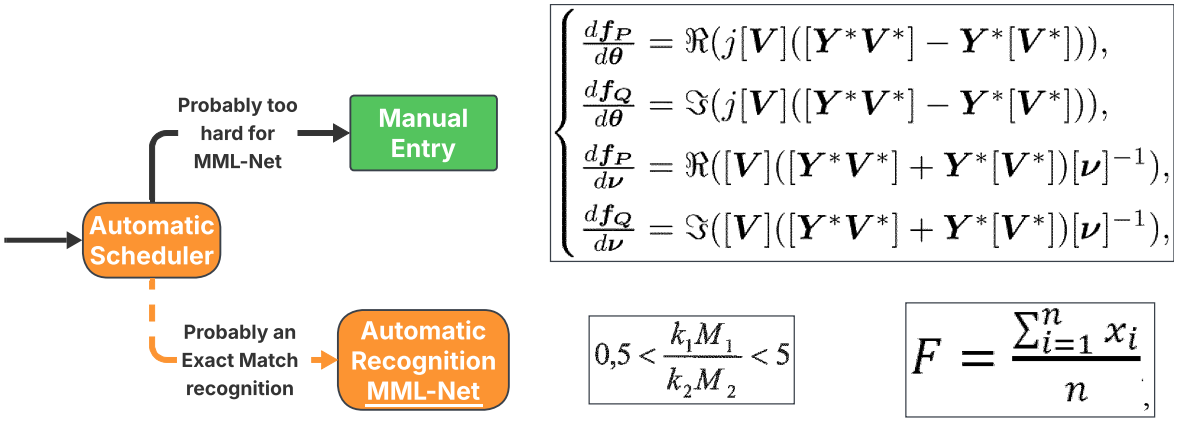


Figure 7.1: Illustration of the automatic scheduler goal.

Its purpose is to prevent expressions that are unlikely to be correctly recognized by MML-Net from entering the automatic pipeline. Such expressions can instead be routed directly to human annotators, avoiding both unnecessary OCR processing and the additional effort required to identify and correct OCR errors.

The problem is formulated as a binary classification task, where the target indicates whether the OCR output reaches Exact Match with the reference MathML. Three models are evaluated for this task. The first is a feature-based Random Forest classifier using simple visual properties extracted from the input image. The second is a pretrained ResNet18 classifier fine-tuned on our data. The third is a Vision Transformer (ViT-B/16), also initialized with pretrained weights and fine-tuned on our data.

For each input expression, the three models produce a score between 0 and 1 estimating the likelihood that MML-Net will produce an Exact Match. This score allows the scheduler to operate with a decision threshold selected according to the requirements of the production pipeline. A high threshold can be used when only highly reliable OCR predictions should be processed automatically, while a lower threshold allows a larger proportion of expressions to be processed automatically at the cost of accepting more potentially incorrect predictions.

For an input mathematical expression image x, the target variable is defined as

$$y = \begin{cases} 1 & \text{if the MML-Net output reaches Exact Match,} \\ 0 & \text{otherwise.} \end{cases}$$

The scheduler estimates the value $p(y = 1 \mid x) \in [0, 1]$. A decision threshold τ can then be applied to this score:

$$\hat{y} = \begin{cases} 1 & \text{if } p(y = 1 \mid x) \geq \tau, \\ 0 & \text{otherwise.} \end{cases}$$

If the predicted probability is above the threshold τ, the expression is considered suitable for automatic transcription and is sent to MML-Net. Otherwise, it is considered too difficult for reliable automatic recognition and is sent to the existing manual transcription workflow. The choice of τ determines how selective the scheduler is: a higher threshold sends fewer expressions to MML-Net, while a lower threshold sends more expressions to the automatic pipeline.

7.1.2 Construction of the Scheduler Dataset

The scheduler dataset is constructed from the predictions of MML-Net on the PatentME-OCR test dataset, presented in Section 4.4. The evaluation of MML-Net on these samples provided, for each expression, both the input image and the corresponding OCR

output, as well as the MathML Exact Match metric. It provides a direct binary target indicating whether MML-Net produced a fully correct transcription. This information is used to construct the target labels for the scheduler.

An expression is assigned to the positive class when the MML-Net OCR prediction exactly matches the ground-truth MathML (Exact Match), and to the negative class otherwise. This results in 16,955 positive examples (42.87%), for which MML-Net produces an Exact Match, and 22,597 negative examples (57.13%) for which it does not. The resulting dataset contains 39,552 mathematical expression images. It is divided into training, validation, and test sets using an 80%, 10%, 10% split, while preserving the class distribution across the three subsets and ensuring that expressions from the same patent remain in the same split.

With this data, the scheduler does not learn to model the intrinsic difficulty of a mathematical expression independently of the OCR model. It is taught to estimate whether our specific MML-Net model is likely to recognize the expression correctly.

7.1.3 Visual Feature Analysis

Before training the scheduler models, we analyze the correlations between the visual properties of the input images and MathML Exact Match. Taking inspiration from textual OCR performance estimation methods [195], which often rely on image quality indicators such as blur [196, 197], our objective is to determine whether simple image characteristics contain information about the difficulty of MER and to identify which visual properties are most associated with recognition errors in MML-Net. Since our input images are binary, the analysis focuses on characteristics that can be derived directly from their structure and pixel distribution rather than on conventional image-quality measures such as blur.

A set of handcrafted features is extracted from each image. These features describe four main properties: image geometry, pixel intensity, connected components, and horizontal and vertical spatial projections.

The original image is first used to compute:

- image height (H), width (W), and aspect ratio (AR);
- image surface area (A), its logarithm ($\log(A)$), and square root ($\sqrt{A}$);
- mean pixel intensity (μ_I);
- number of connected components (NCC), the mean (μ_{CA}) and standard deviation (σ_{CA}) of their surface areas;
- standard deviation of the horizontal (σ_H) and vertical (σ_V) pixel intensity projections.

The same intensity-, component-, and projection-based features were computed after resizing every image to 256 × 256 pixels. This produces a total of 18 visual features.

Pearson correlations were used to measure the relationship between these features and two OCR evaluation metrics. We considered both MathML Exact Match and MathML Edit Score, for comparison with both a binary and a continuous target. An Exact Match value of 1 indicates perfect recognition and a value of 0 indicates that at least one error is present. Similarly, an Edit Score of 1 indicates perfect recognition and lower scores indicate increasing differences between the prediction and the reference.

In the resulting correlation matrix shown in Figure 7.2, positive values indicate that higher feature values are associated with better recognition performance, while negative values indicate the opposite. Values close to zero indicate little or no relationship, whereas values closer to −1 or 1 indicate stronger relationships.

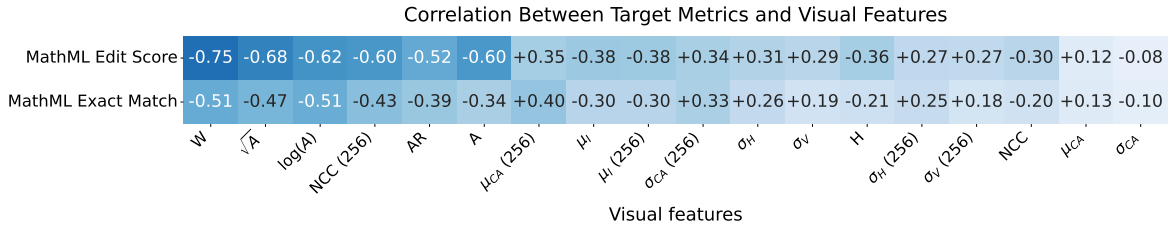


Figure 7.2: Correlations between the extracted visual features and the OCR evaluation metrics.

The correlation heatmap shows that simple geometric and structural properties contain useful information about OCR difficulty. Image width W has the strongest correlation with Exact Match (-0.51), while the logarithm and square root of the image area are also strongly correlated. Their stronger correlation compared with the raw area suggests that reducing the effect of very large values with $\log(\cdot)$ or $\sqrt{\cdot}$ provides a better representation of image size for the linear correlation analysis. Overall, larger expressions are associated with lower Exact Match performance from MML-Net.

Connected-component and projection features also provide additional information about expression complexity. A higher number of connected components (NCC) and larger mean component area (μ_{CA}) are associated with lower recognition performance, whereas higher variability in component sizes (σ_{CA}) and pixel projections (σ_H and σ_V) is associated with better performance. The negative correlation of μ_{CA} and the positive correlation of σ_{CA} may indicate that expressions with many components of similar size are more difficult to recognize. An example of such an expression could be the large matrix shown in the top-right example of Figure 7.1. The relative importance of these features also depends on whether they are computed on the original image or on the 256 × 256 resized image. Overall, the analysis shows that OCR Exact Match is related to several visual characteristics of the expression, supporting the use of these features for the scheduler classification task.

7.1.4 Evaluated Model Architectures

Three classification approaches were evaluated. They were selected to provide different levels of complexity, from a simple feature-based model to pretrained deep visual models.

Random Forest on Handcrafted Features

The first approach uses the Random Forest classifier on the 18 handcrafted image features described in the previous section. It is a lightweight baseline that does not require a deep neural network. To determine the best number of features to use, we performed a feature-count sweep. For each value of k from 1 to 18, the top-k features with the highest absolute Pearson correlation with the Exact Match target were selected, following the order in the correlation analysis shown in Figure 7.2. A Random Forest containing 100 decision trees was then trained using these features. Validation and test accuracy and F1 scores were measured for each value of k. The results are reported in Table 7.1 and Figure 7.3.

Table 7.1: Random Forest performance for different numbers of selected visual features.

Number of features	Validation Accuracy	Validation F1	Test Accuracy	Test F1
1	76.21%	78.67%	77.85%	80.12%
2	78.41%	79.90%	78.94%	80.26%
3	78.38%	79.88%	78.94%	80.26%
4	81.67%	81.80%	81.72%	81.89%
5	81.49%	81.53%	81.97%	82.15%
10	85.51%	85.53%	86.47%	86.62%
14	87.31%	87.36%	88.02%	88.14%
18	87.86%	87.94%	88.19%	88.32%

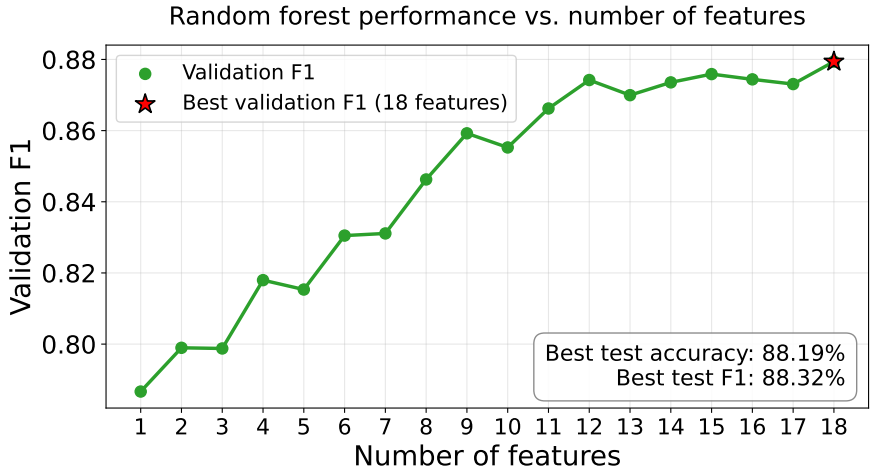


Figure 7.3: Random Forest performance as a function of the number of selected visual features.

The feature sweep shows a clear improvement in validation F1 as more visual features are introduced, increasing from 78.67% with a single feature to 81.80% with 4 features and 87.94% with all 18 features. The configuration using all 18 features provides the best validation performance. On the test set, this configuration achieves an F1 score of 88.32% and an accuracy of 88.19%, and is kept for comparison with the other scheduler models introduced below.

The results also show that features with relatively weak individual Pearson correlations can still contribute to the classification performance when combined with other features. They capture complementary aspects of expression difficulty that are not fully reflected by their individual linear correlations with Exact Match. Since these handcrafted features are inexpensive to compute, using the complete feature set adds little additional computational cost.

ResNet18

The second approach uses a ResNet18 convolutional neural network initialized with ImageNet pretrained weights. ResNet18 is a compact convolutional architecture that learns visual patterns directly from the expression image, without relying on handcrafted features. The final classification layer is replaced by a two-class linear layer. All 11,177,538 model parameters are trainable.

ViT-B/16

The third approach uses a Vision Transformer, ViT-B/16, also initialized with ImageNet pretrained weights. This architecture was selected because Transformer-based visual encoders have shown good performance on mathematical expressions in previous experiments, although they are more computationally demanding than the Random Forest and convolutional approaches. The final classification head is replaced by a two-class linear layer. All 85,800,194 model parameters are trainable.

Training and Evaluation Protocol

Both deep models use their standard ImageNet preprocessing, including resizing and normalization according to the corresponding pretrained weights. They are trained on the scheduler training dataset for 15 epochs using the Adam optimizer, with a batch size of 16 and the cross-entropy loss. The initial learning rate is set to 10^{-3}.

The predicted probabilities of the Exact Match class are saved for each validation and test sample. Precision and recall are evaluated over a range of thresholds and visualized using precision–recall curves. The validation set is used to select the classification threshold τ that maximizes recall while maintaining a precision of at least 95%. This reflects the intended use of the scheduler: we want to identify expressions that can be

confidently processed automatically while selecting as few hard expressions as possible. The test set remains isolated during model selection and is used only for the final evaluation.

7.1.5 Comparison of the Three Approaches

The three trained models are first evaluated on the validation set, which is used to select the decision threshold τ. The threshold is chosen to maximize recall while maintaining a target precision of 95%. This selection is performed from the validation precision–recall curves shown in Figure 7.4.

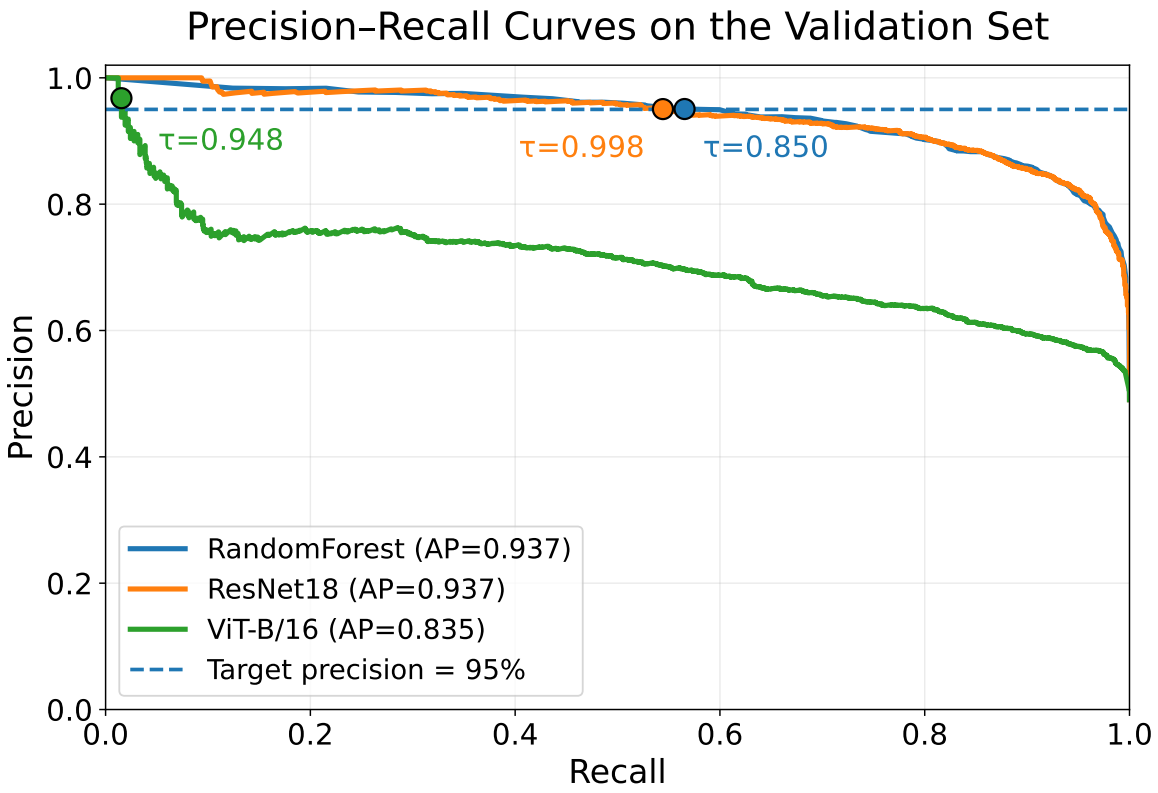


Figure 7.4: Precision–recall curves on the validation set for the three scheduler models.

The PR curves show a clear difference between the three approaches. ResNet18 and Random Forest achieve similar average precision values of 0.937, whereas ViT performs worse, with an average precision of 0.835. The selected thresholds are $\tau = 0.850$ for Random Forest, $\tau = 0.998$ for ResNet18, and $\tau = 0.999$ for ViT. The very high thresholds selected for the two neural models indicate that their predictions must be filtered very aggressively to reach the target precision of 95%. These conservative operating points come at the cost of lower recall.

The selected thresholds are then fixed and applied to the predictions on the test set. The resulting confusion matrices are shown in Figure 7.5. The corresponding test precision (P) and recall (R) are reported above each confusion matrix.

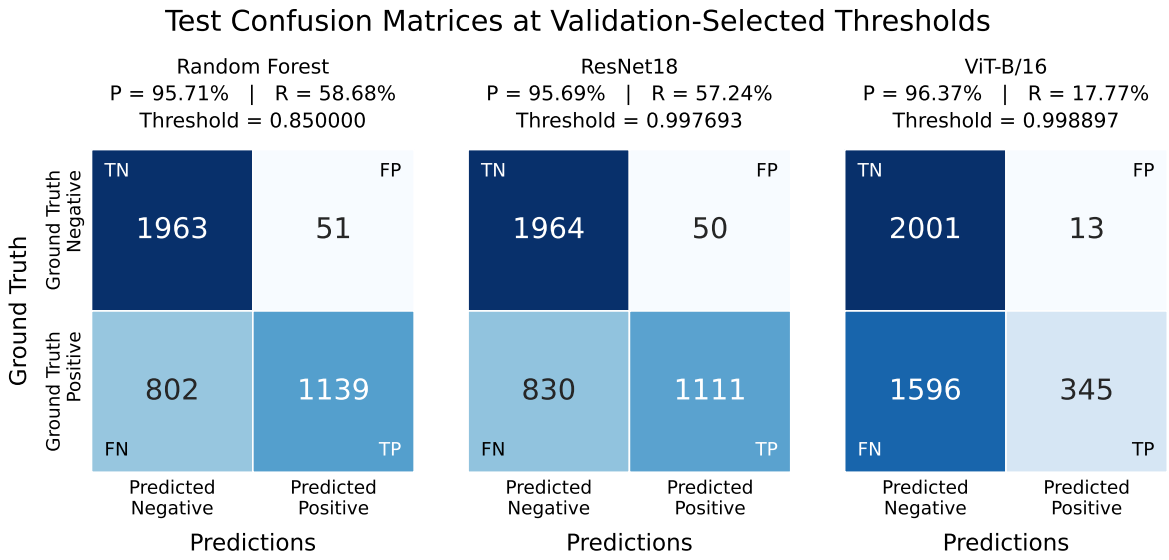


Figure 7.5: Confusion matrices on the test set using the decision thresholds selected on the validation set. Test precision and recall are reported above each matrix.

On the test set, Random Forest achieves a precision of 95.71% and a recall of 58.68%. ResNet18 obtains a similar precision of 95.69%, with a recall of 57.24%. ViT achieves the highest precision of the three models, at 96.37%, but its recall drops to only 17.77%. The corresponding coverage values, which represent the total number of expressions that are predicted positive by the models and sent to MML-Net for recognition, are 30.1% for Random Forest, 29.4% for ResNet18, and only 9.1% for ViT. Even though the three models satisfy the target precision, only a limited portion of expressions are classified as suitable for processing by MML-Net. This limitation is particularly pronounced for ViT and results in very low coverage.

Overall, Random Forest provides the best trade-off at the selected 95% precision operating point. In addition, it is considerably lighter computationally than the two neural approaches, making it particularly attractive for large-scale deployment of the scheduler.

The 95% precision operating point should, however, not be considered the only possible operating point. The decision threshold τ controls the trade-off between precision and recall and can be adjusted according to the intended use of the scheduler. A lower threshold would increase recall and allow a larger proportion of difficult expressions to be passed to the OCR system, at the risk of potentially generating more incorrect predictions to be caught by downstream verifications.

The comparatively weak performance of ViT may be related to the experimental setting. ViT-B/16 is a larger and more data-hungry architecture than the other approaches, and its performance may require a larger training set and longer training time. In contrast, the lightweight and fast Random Forest model is particularly well suited to the requirements of the scheduler's use case.

7.1.6 Discussion on the Scheduler

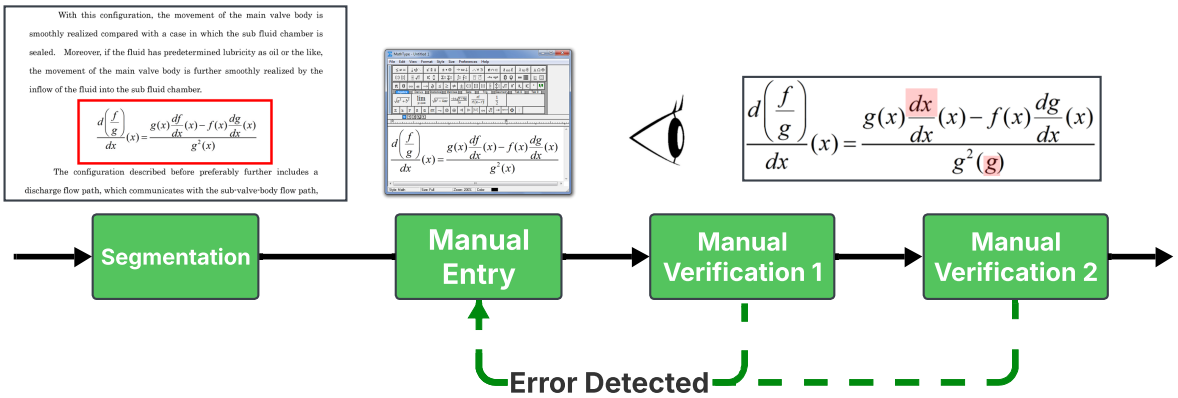
These results show that relatively simple visual information can already provide a strong estimate of OCR difficulty. The Random Forest requires only the extraction of a small number of image statistics, whereas ResNet18 and ViT require a complete forward pass through a pretrained image encoder. However, the feature-based approach is based on handcrafted visual descriptors and may be affected by biases introduced during feature design. It is also trained to predict the success or failure of the MML-Net model. A change in the OCR architecture or training data would most likely require retraining the scheduler.

Nevertheless, its low computational cost and competitive predictive performance make it suitable when the scheduler must be applied to a large number of expressions. In addition, the scheduler could be operated with a lower threshold τ if a higher coverage is preferred, allowing more difficult expressions to be processed automatically while relying on the downstream verifications to identify recognition errors.

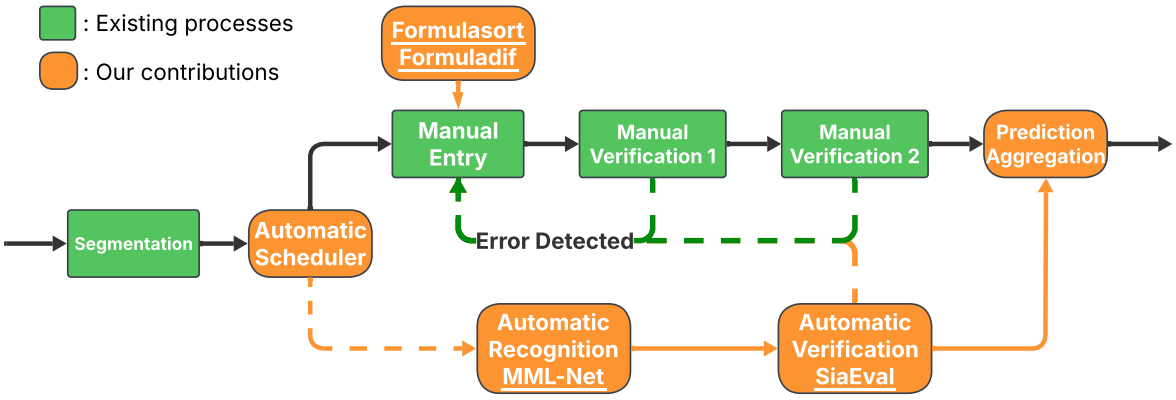
7.2 Evaluation of the Semi-Automatic Processing Pipeline

This section integrates the contributions developed throughout this thesis into an improved semi-automated pipeline for the transcription of mathematical expressions at Luminess. The objective of this pipeline is to increase the level of automation, reduce transcription errors, and reduce the repetitive workload of human operators.

Figure 7.6 compares the current Luminess production workflow with the proposed integrated pipeline. The proposed workflow does not aim to completely replace human transcription. Instead, it introduces several automatic components that identify expressions suitable for automatic recognition, process them using MML-Net, verify the resulting predictions, and provide additional support to operators for the expressions that instead go through the manual workflow.



(a) The existing Luminess pipeline for the manual transcription of mathematical expressions.



(b) Proposed pipeline combining the contributions developed throughout the thesis.

Figure 7.6: Overview of the existing and proposed mathematical expression recognition pipelines.

7.2.1 Proposed Workflows

To illustrate how the different components could interact in practice, this section considers a hypothetical batch corresponding to the 3153 mathematical expressions of the PatentME-OCR test set. The results are obtained by combining the performances measured independently for each component and should not be interpreted as a strict evaluation of the complete pipeline. This simulation is used to compare two operating modes. Mode A maximizes the reduction in human processing time, and mode B maximizes the quality of the final transcriptions. To illustrate this section with the actual number of processed expressions, all values are rounded to the nearest unit.

Mode A: Maximum Reduction of Human Processing Time

In this mode, the objective is to minimize human intervention. The scheduler routes each expression either to the manual branch or to the automatic branch. Expressions sent to the manual branch are transcribed by human operators, while expressions selected for the automatic branch are processed exclusively by MML-Net. SiaEval then evaluates the resulting automatic predictions and either accepts them as final outputs or routes them to human verification. At the end of the process, there is no prediction aggregation. We either use the prediction from the manual branch, or the one from the automatic branch. This mode is illustrated in Figure 7.7.

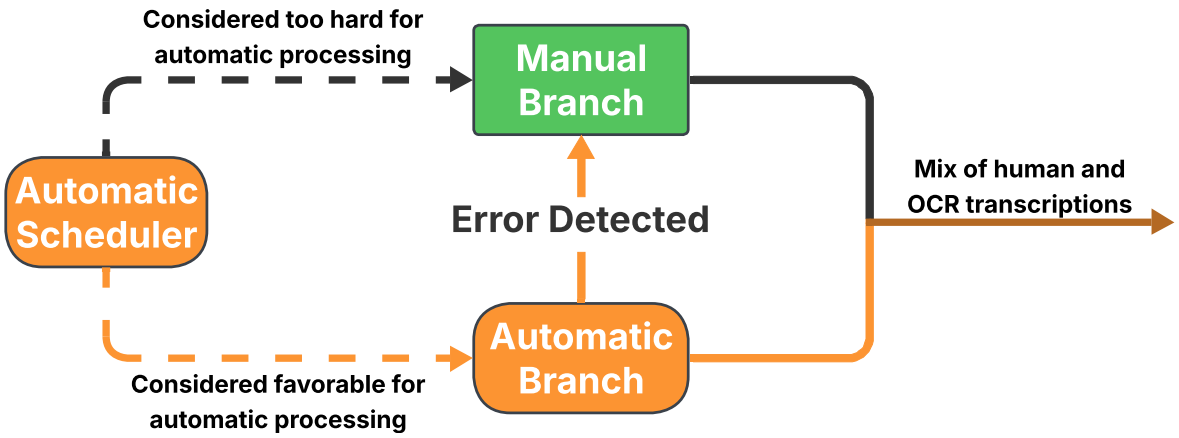


Figure 7.7: Mode A : minimize the required human intervention.

Based on the scheduler results (Figure 7.5), 1963 expressions would be directly routed to manual processing, while 1190 would be processed by MML-Net. Among these 1190 expressions, approximately 1139 would be correctly recognized and 51 incorrectly recognized. This corresponds to an effective Exact Match accuracy of $\frac{1139}{1190} \approx 95.71\%$. This value is considerably higher than the overall 47.5% Exact Match Ratio obtained by MML-Net in Section 4.4.4, as the scheduler filters out expressions for which MML-Net is more likely to fail.

SiaEval would then evaluate the 1190 automatic predictions, including the 51 errors produced by MML-Net. At the more lenient operating point using $\tau = 0.098$, giving a true negative rate of 92.45% and a recall of 73.72%, approximately 844 predictions would be automatically accepted by SiaEval, including 4 erroneous predictions. Relative to the initial batch, this corresponds to a residual error rate of approximately $\frac{4}{3153} \approx 0.13\%$, or an estimated quality of approximately 99.87% under our assumptions. The remaining 346 expressions would be rejected and redirected to human verification, including 299 predictions that were actually correct.

At the double manual verification stage, all manually transcribed expressions and all expressions rejected by SiaEval must be verified. This results in $1963 + 346 = 2309$ human verification operations.

Formulasort also reduces the time required for manual transcription. Its effect on transcription quality has not been quantitatively evaluated, but it was estimated in Section 6.2 that copy-paste could be used for 8.83% of the expressions instead of transcribing them from scratch.

We use a simplified cost model in which a manual transcription costs 2 time units, a copy-and-paste transcription costs 1 time unit, and a verification costs 1 time unit. These values were discussed with the operators involved in the production workflow, who considered them to provide a reasonable average estimate of the relative time of these operations. With this model, the fully manual workflow requires:

$$\underbrace{(2 + 2 \times 1)}_{\substack{\text{one manual transcription} \\ \text{+ two manual verification}}} \times \underbrace{3153}_{\text{number of expressions}} = 12612$$

time units for the complete batch.

For Mode A, the estimated human processing time is:

$$\underbrace{(2(1 - 0.0883) + 1 \times 0.0883)}_{\substack{\text{average transcription cost} \\ \text{with Formulasort}}} \times \underbrace{1963}_{\substack{\text{manual} \\ \text{+transcriptions}}} + \underbrace{2 \times \underbrace{(1963 + 346)}_{\substack{\text{manually transcribed} \\ \text{+ rejected by SiaEval}}}}_{\text{double verification}} = 8371$$

time units. This represents a reduction of $12612 - 8371 = 4241$ time units, corresponding to a 33.63% reduction in human processing time. Knowing the workload of mathematical transcription represents 230.5 hours of manual processing per week, this corresponds to a reduction of approximately 77.5 hours per week.

This mode prioritizes processing-time reduction. Its main limitation is that automatic predictions accepted by SiaEval do not benefit from an independent human transcription. In particular, the approximately 4 errors not detected by SiaEval would remain in the final patent output in this simplified scenario, since no prediction comparison between

human and automatic transcriptions is performed with this mode.

Mode B: Maximize the Transcription Accuracy

The second mode introduces an extra layer of human supervision. The scheduler still separates the input expressions into a manual branch and an automatic branch, but expressions selected for the automatic branch are also manually transcribed. This provides both an independent human reference and an automatic prediction for a subset of the expressions, allowing us to perform prediction comparison at the end of the process. This mode is illustrated in Figure 7.8.

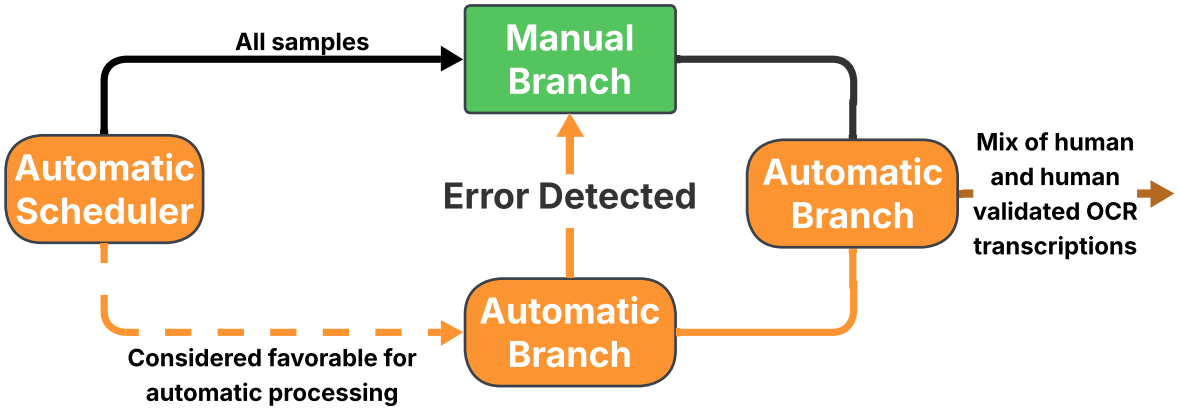


Figure 7.8: Mode B : maximize the transcription accuracy.

In this configuration, all 3153 expressions are routed to manual processing, with 1190 of them additionally processed by MML-Net. For these 1190 expressions, MML-Net produces the same approximately 1139 correct and 51 erroneous predictions as in Mode A.

SiaEval would then reject 346 expressions for human verification, including 299 predictions that were actually correct, and accept 844 predictions, including the 4 erroneous predictions. In addition, the prediction aggregator can compare the independent human transcriptions with the automatic predictions. Assuming that the aggregator identifies disagreements for the 4 cases where the automatic prediction is incorrect, an additional 4 human verification operations would be required.

Under the same simplified cost model, the initial human processing and verification operations require:

$$\underbrace{(2(1-0.0883)+1\times 0.0883)}_{\substack{\text{average transcription cost}\\ \text{with Formulasort}}}\times \underbrace{3153}_{\substack{\text{manual}\\ \text{transcriptions}}} + \underbrace{2\times \underbrace{(3153+346)}_{\substack{\text{manually transcribed}\\ \text{+ rejected by SiaEval}}}}_{\text{double verification}} + \underbrace{4}_{\substack{\text{additional}\\ \text{verifications}}} = 13030$$

time units.

Compared with the fully manual workflow, this increase of $13030 - 12612 = 418$ time units corresponds, relative to the 230.5 hours of manual processing per week, to an increase of only 3.31% or 7.6 hours per week in human processing time.

As a result, the 1190 expressions processed by the automatic branch benefit from both an independent human transcription and an automatic recognition. This provides an additional mechanism for detecting errors in either transcription: a disagreement between the two outputs can trigger a final human verification. Under the assumptions of this simplified scenario, this additional layer of supervision could theoretically improve the final quality beyond that of the current workflow. Performing this extra verification for 1190 expressions costs here 418 extra time units, and would have cost 1190 time units if performed manually according to our simulation.

The estimated processing time, weekly workload, and expected quality of the three configurations are summarized in Table 7.2. The values in parentheses indicate the gain or loss relative to the fully manual workflow.

Table 7.2: Theoretical comparison of the proposed processing workflows.

Method	**Time (units)**	**Time (h/week)**	**Expected quality**
Fully manual	12612	230.5	99.99%
Mode A	8371 (−33.63%)	153 (−77.5 h)	99.87% (−0.12 pp)
Mode B	13030 (+3.31%)	238.1 (+7.6 h)	99.99 + %

7.2.2 Discussion and Limitations of the Integrated Evaluation

Overall, the two operating modes illustrate the trade-off between processing efficiency and quality assurance. Mode A minimizes human intervention and provides a theoretical reduction in human processing time, but accepts the risk of undetected errors in the automatic branch. Mode B performs a human transcription for all expressions while adding automatic recognition, SiaEval, and prediction aggregation for a subset of the expressions, resulting in a slightly higher human processing cost but providing additional methods for detecting and resolving errors.

The complete pipeline has not yet been deployed or evaluated end-to-end in the Luminess production environment. The quantitative scenario presented above should be interpreted as a simulation of performance rather than as an evaluation of the complete system. The numerical results combine the individual performances of the different systems and do not account for unforeseen interactions between them. There is also a risk of distribution shift between the experimental and production distributions, and variations in human operator behavior.

A robust evaluation should process the same selection of patents through both the existing and the two proposed workflows under real conditions. Relevant measurements

should include total processing time, time spent at each stage, routing decisions, automatic predictions accepted or rejected, and final transcription accuracy. This paired evaluation would directly quantify the time and quality differences introduced by the proposed workflow against the manual pipeline. Human operator feedback is also necessary to assess the practical impact of Formulasort, Formuladif, and prediction aggregation. Their benefits depend heavily on how operators interact with the interface, how quickly they verify suggested transcriptions, and whether the proposed parameters are appropriate under real working conditions.

Operator training is another important consideration for deployment. If Mode A is introduced directly, new operators could be exposed only to the most difficult and ambiguous expressions that MML-Net cannot recognize. This could make the manual workload to appear more complex and potentially increase fatigue and frustration. A progressive deployment strategy could reduce this effect. Mode B could first be introduced to familiarize operators with the new tools and workflow, followed by a gradual transition toward Mode A as operators become familiar with the system.

We could also assign new operators a larger proportion of easier expressions to help them become familiar with the transcription process, before progressively handling the more difficult expressions routed to the manual branch in Mode A. In practice, increasingly large batches of expressions could be routed exclusively to MML-Net while monitoring quality and operator feedback.

7.3 Conclusion

This chapter presented an integrated workflow combining the contributions developed throughout this thesis into a coherent semi-automated processing architecture. The scheduler provides a routing decision for expressions according to their expected OCR prediction accuracy. MML-Net automatically processes expressions considered suitable for recognition, while SiaEval provides an additional verification layer. Remaining expressions are processed manually, with Formulasort reducing repetitive transcription work and Formuladif facilitating the identification of subtle visual differences. Finally, the prediction aggregation module compares the available human and automatic predictions to identify potential disagreements requiring further verification.

Under the simplified assumptions of the quantitative scenario, Mode A reduces estimated human processing time by 33.63%, corresponding to approximately 77.5 hours saved per week, while reaching an estimated quality of approximately 99.87%. Mode B increases estimated human processing time by only 3.31%, or approximately 7.6 additional hours per week, while providing additional independent verification mechanisms that could potentially reduce the residual error rate and raise the final quality beyond 99.99%.

The complete workflow has not yet been evaluated end-to-end in the production environment. A controlled comparison of identical patents processed through the current and proposed workflows, combined with quantitative measurements and operator feedback, will be necessary to determine the actual gains and calibrate the different components before deployment.

Chapter 8

General Conclusion and Perspectives

8.1 Industrial Context and Research Problem

This thesis investigated how mathematical expressions in real-world technical documents can be recognized and selectively processed with a high level of reliability, in order to enable the indexing and retrieval of mathematical information from large document databases.[2]

In particular, this thesis investigated the pipeline used to process *patent applications* at the EPO, as part of the industrial partnership with Luminess. More specifically, it focused on the scientific problem of MER, a research topic that has been studied for several decades and that has made significant progress with the emergence of deep learning methods and end-to-end architectures.

Mathematical expressions in patent applications have strong visual heterogeneity due to variations in printing quality, resolution, font, typographic style, and the various artifacts that can occur in real-world documents. They must also be converted into MathML so that they can be integrated into the final XML representation. This context differs from the expressions often considered in the literature, which are clean or synthetically generated expressions from LaTeX.

The target accuracy of the processing pipeline for patent applications is 99.99%, corresponding to the proportion of expressions that must be recognized exactly without errors. Currently, this level of accuracy is achieved through a fully manual process in which expressions are transcribed and verified twice by human operators. Such a high level of accuracy is difficult to achieve with a recognition model alone and requires a complete processing pipeline with additional verification mechanisms, such as confidence scores, human validation, and other quality-control measures.

The objective of this thesis was to evaluate existing MER models, train models for this specific task, and also propose a processing pipeline integrating pre- and post-processing around the recognition model, with the possibility of involving human operators when required. The goal of these extra steps is to determine which expressions can be reliably processed automatically and which expressions should be verified or processed by a human operator, with the aim of reducing human processing time, improving quality, and reducing the cognitive workload of operators.

8.2 Answer to the Research Questions

8.2.1 Capabilities and Limitations of Current MER Methods

RQ1: *What are the capabilities and limitations of current MER methods when applied to real-world patent documents?*

This first research question is answered by the state-of-the-art presented in Chapter 2, and is supported by the evaluations conducted on the PatentME-OCR dataset in Section 4.4.

The state of the art investigated MER along three complementary dimensions: models, datasets, and evaluation metrics.

Regarding models, the literature shows a gradual transition from traditional approaches based on pipelines separating symbol recognition from structural analysis to end-to-end architectures based on deep learning. Encoder–decoder architectures based on Vision Transformers currently constitute a dominant family of models for MER. Recent progress in general-purpose Vision-Language Models in document understanding tasks has not yet translated into comparable performance on the specific task of MER, particularly on real-world patent expressions.

Datasets have evolved towards very large-scale collections mainly generated from LaTeX, providing the amount of data required to train modern deep models. However, they often lack variations in fonts, backgrounds, rendering quality, and degradations and rely on clean synthetic expressions. Another limitation concerns the target representation. Most existing datasets use LaTeX, while structurally richer representations such as MathML or Label Graphs remain less commonly used for MER.

Evaluation also presents several limitations. Text-based metrics for LaTeX, such as Exact Match and edit distances, measure the correspondence between predicted and reference markups, but can be sensitive to differences in representation between mathematically or visually equivalent expressions. Visual and structural metrics provide more information, and embedding-based approaches offer another interesting direction for measuring semantic or multimodal similarity. However, no single metric can capture all relevant dimensions of recognition quality.

To further investigate these limitations on patent expressions, a selection of five models from the literature was evaluated on the PatentME-OCR dataset introduced in Section 4.4, which provides mathematical expression images paired with MathML references from patent applications. The best-performing previous model achieved only 21.8% Visual Exact Match, as shown in the rightmost column of Figure 4.10.

Overall, current MER architectures have achieved strong performance on existing benchmarks, but the datasets and evaluation procedures used to develop them remain limited in their ability to reflect real-world patent documents.

8.2.2 Measuring Quality and Reliability

RQ2: *How can the quality and reliability of MER predictions be measured, both during model evaluation and in production?*

Two contributions address this question: PiE-MER in Chapter 3 and SiaEval in Chapter 5. They propose two different approaches to evaluate the quality and reliability of MER systems.

The PiE-MER results show that an evaluation based on a single metric or modality can provide an incomplete characterization of model performance, and may even lead to different conclusions depending on the evaluation measure considered. A prediction may be considered incorrect according to a textual metric while exhibiting high visual or structural similarity to the reference. PiE-MER addresses this limitation by considering several modalities for evaluation, comparing mathematical expressions at the LaTeX, MathML, image, and Label Graph levels. This multimodal evaluation makes it possible to better characterize the errors produced by MER systems and avoids reducing recognition quality to an exact match between two sequences.

Such reference-based evaluation requires access to a ground-truth reference, which is unavailable in production. To evaluate a model in production, we can only use the input image and the model prediction, so metrics that require the ground-truth cannot be used to evaluate the quality of the predictions. SiaEval addresses this limitation by introducing a reference-free post-OCR verification framework that compares the embeddings of the input image and the rendered model prediction with a Siamese network. It outputs a confidence score that can be used to accept or reject the OCR predictions.

Together, these two contributions make it possible to evaluate the quality of MER systems both during model development, when reference annotations are available, and for estimating prediction reliability during operational use, when only the input image and the model prediction are available.

8.2.3 The Role of Large-Scale Domain-Specific Data

RQ3: *How can large-scale technical document databases be used to improve automatic MER?*

This research question is addressed through the creation of two domain-specific datasets, PatentME-600k and PatentME-OCR, using the large-scale patent collection available on the EPO online portal. PatentME-600k contains more than 600,000 image–MathML pairs and was designed for training recognition models, while PatentME-OCR was designed as a test resource for evaluating their performance on patent mathematics. A statistical analysis of these datasets was conducted to characterize the diversity of the mathematical expressions encountered in patent applications, in terms of visual

characteristics and MathML complexity.

To make these data usable for training a new MER model, we introduced a dedicated MathML tokenizer. We used it to train MML-Net, a ViT-based encoder–decoder specifically designed for MER in patent applications. MML-Net was compared with several models from the literature and achieved 54.89% MathML Exact Match and a normalized Levenshtein distance of 0.0631 on the PatentME-600k validation set.

These results show that domain-specific training is important for reliable MER performance on patent mathematics. Models developed and evaluated on existing public datasets may not transfer reliably to the heterogeneous expressions of patent applications. The creation of large-scale domain-specific datasets is a key step towards developing MER systems that can meet the requirements of industrial applications.

8.2.4 Assistance to Human Operators

RQ4: *How can automatic systems assist human operators by reducing processing time, errors, and cognitive load?*

The manual transcription and verification process was studied through discussions with operators performing manual transcription in Romania, as well as direct observation of the process on site. The difficulties associated with the transcription and verification of mathematical expressions were identified and reported in Section 6.2.

Based on these observations, two contributions were developed to assist operators without replacing the existing human workflow. Formulasort changes the order in which mathematical expressions are presented to operators by grouping similar expressions together. This facilitates the use of the copy-paste tool of the transcription interface, and aims at reducing unnecessary task switching, processing time, and cognitive load.

Formuladif provides an automated visual comparison of successive mathematical expressions, highlighting subtle differences between expressions that may be overlooked. This facilitates the verification process and helps reduce transcription errors, particularly when the transcription is performed rapidly and relies heavily on copy-paste.

Formulasort and Formuladif illustrate how automatic assistance can improve an existing human-in-the-loop workflow by simplifying repetitive tasks, facilitating verification, and helping operators focus their attention on subtle differences.

8.2.5 Conditions for Reliable Automation

RQ5: *Under which conditions can MER be reliably automated, and when is human intervention still required?*

The answer to this research question relies on the combination of the Scheduler from Section 7.1, MML-Net from Chapter 4, and SiaEval presented above.

A first requirement for reliable automation is a recognition model adapted to the target domain. The experiments conducted with MML-Net demonstrate the importance of training on data representative of the mathematical expressions encountered in patent documents. MML-Net achieves a Visual Exact Match for approximately half of its predictions. Although this performance is above the state of the art for this task, it also shows that a recognition model alone is not sufficient to guarantee reliable automation, as a large proportion of predictions still contain errors.

This is why this thesis introduces methods for estimating and verifying the reliability of recognition predictions. The Scheduler operates before recognition and uses the visual characteristics of mathematical expressions to estimate the probability that it will be correctly recognized by MML-Net. It can identify potentially difficult cases and route expressions either towards automatic transcription or towards the existing human transcription process. This Scheduler can be operated at different thresholds to adjust the balance between the level of automation and the error risk.

SiaEval provides a complementary verification mechanism after recognition. It can identify predictions that are likely to be incorrect and redirect them to human verification. SiaEval can also be operated at different thresholds to balance the level of automation and potential error rate.

The combination of automated recognition, prediction validation, and the existing human processing pipeline for uncertain cases makes it possible to control the level of automation while retaining human intervention for expressions that cannot be reliably handled automatically.

8.3 Proposed Semi-Automatic Pipeline

Chapter 7 integrates the different contributions into a semi-automatic processing pipeline for reliable MER. The proposed approach extends the existing manual workflow with automatic recognition, difficulty estimation, and prediction verification mechanisms. The resulting pipeline is illustrated in Figure 8.1. Two operating modes were proposed: **Mode A** aims to minimize human processing time, while **Mode B** aims to maximize the reliability of the final output.

A simulated evaluation of the two modes on the complete pipeline was conducted by combining the results obtained in the previous chapters. The comparison with the fully manual workflow is reported in Table 8.1. Mode A reduces the estimated processing time by 33.63%, while maintaining an expected quality of 99.87%. Mode B increases the estimated processing time by 3.31% compared with the fully manual workflow, but integrates an extra automatic verification step to reduce the residual error rate.

In practice, the optimal configuration will depend on the target balance between automation and reliability, as well as on the thresholds selected for the different modules.

Table 8.1: Theoretical comparison of the proposed processing workflows.

Method	Time (units)	Time (h/week)	Expected quality
Fully manual	12612	230.5	99.99%
Mode A	8371 (−33.63%)	153 (−77.5 h)	99.87% (−0.12 pp)
Mode B	13030 (+3.31%)	238.1 (+7.6 h)	> 99.99%

Further evaluation in real production conditions should identify the most appropriate configuration and incorporate operator feedback on the usefulness and effects of the different components.

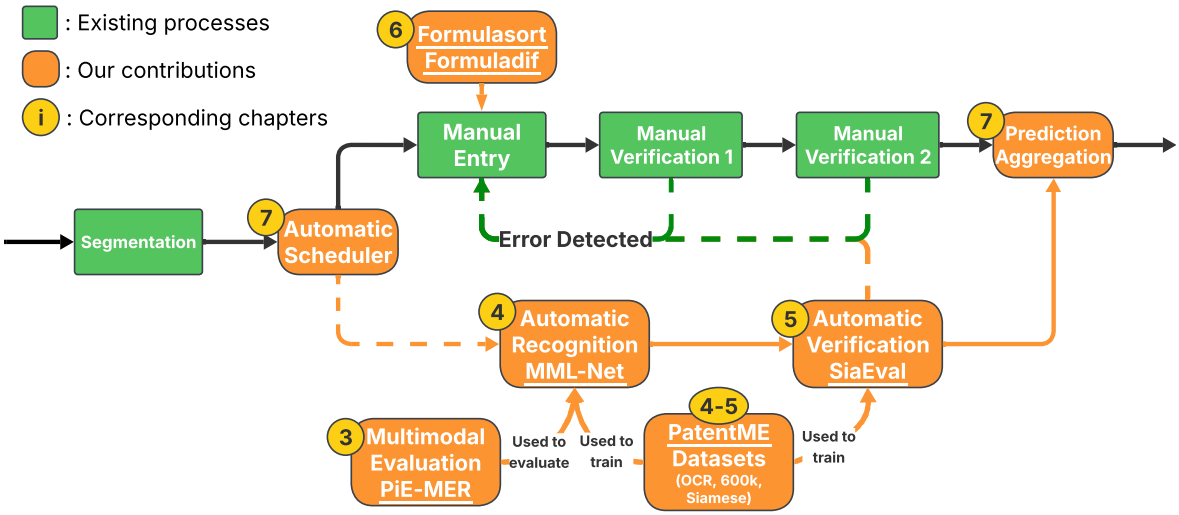


Figure 8.1: Proposed pipeline combining the contributions developed throughout the thesis, with the corresponding chapter numbers indicated on the different components.

8.4 Limitations

Several limitations remain and motivate future research. Most importantly, the proposed semi-automatic pipeline was evaluated through a simulation based on the experimental results obtained in the different chapters. A validation under real production conditions is required to measure actual processing times, error rates, and the impact of each component on the operators' workflow. In particular, the theoretical cost model used to estimate processing time should be confronted with real measurements, taking into account variations in expression complexity and operator behaviour.

Although domain-specific training significantly improved MER, MML-Net does not yet achieve a very high accuracy. Further improvements to MER models could be investigated using approaches identified in the literature, such as more advanced encoders, spotlight and coverage-based attention mechanisms, multi-scale image inputs, automatic segmentation of multi-line expressions, and alternative training strategies such as reinforcement learning. These approaches could be specifically investigated on

mathematical expressions from patent documents.

The Scheduler and SiaEval components also present several limitations. They were both developed and evaluated using predictions from MML-Net. Their ability to generalize to other MER architectures, datasets, error distributions, and production environments remains to be investigated. SiaEval had to be retrained specifically for MML-Net, as the model trained on errors from another MER system did not transfer well to MML-Net errors. Future work should investigate more model-independent approaches that can be transferred across recognition systems. The thresholds used by the Scheduler and SiaEval should also be optimized under real production conditions based on the measured trade-off between automation, processing time, and reliability.

Another limitation concerns the human–machine interaction itself. The proposed assistance tools were designed to reduce repetitive work and facilitate verification, but their impact on operators was not quantitatively evaluated in a controlled production study. Future experiments should measure processing time and error rates, as well as task switching, verification effort, and operator feedback. Such studies would provide a more complete evaluation of the actual value of the proposed pipeline.

8.5 Future Work

Several research directions could further extend this work.

Adaptive and domain-specific MER

A first direction is the development of more specialized MER systems. Recurring fonts, layouts, and document styles could be identified to cluster similar patent applications and develop specialized recognition models adapted to specific document families or visual styles. The textual context surrounding an expression and previously transcribed expressions could also be exploited to resolve ambiguities and improve model predictions.

Reliable and selective automation

Scheduler and SiaEval provide initial mechanisms for deciding which predictions can be processed automatically. Future work could investigate how to combine these approaches with multiple OCR systems and prediction aggregation to make more reliable automation decisions, for example through voting strategies. Improving confidence calibration and routing strategies could also increase the proportion of predictions that can be processed automatically while maintaining the required level of accuracy.

Human-assisted recognition

Human intervention could be made more targeted and informative. Attention maps or intermediate feature maps could be investigated to identify the regions of an expression that are responsible for an incorrect prediction. This information could support targeted human verification by highlighting potentially problematic symbols or regions.

The human annotations available in the production workflow could also be used for active and continual learning. Human corrections could help identify difficult examples and iteratively improve recognition models. Learning from human feedback could also be investigated to incorporate operator corrections into model training. These approaches could be combined with curriculum-learning strategies to progressively expose models to increasingly difficult expressions.

Future work should build on these results by evaluating the proposed approaches in real production conditions and by further improving mathematical processing along three complementary directions: more adaptive and domain-specific recognition models, more reliable and selective automation, and more effective human-assisted recognition. Together, these directions could improve both the accuracy and efficiency of mathematical processing workflows.

The contributions developed during this thesis should also be applied beyond the patent processing use case. The proposed methods could be used on historical mathematical documents and handwritten manuscripts such as Grothendieck's handwritten mathematical manuscripts[1], which present many recognition challenges. Such collections would provide an interesting setting for studying domain adaptation and human-assisted recognition.

[1] See https://webusers.imj-prg.fr/~leila.schneps/grothendieckcircle/unpubtexts.php, accessed 2026-09-09.

Bibliography

[1] Michael K. Buckland. "Emanuel Goldberg, Electronic Document Retrieval, and Vannevar Bush's Memex". In: *Journal of the American Society for Information Science* 43.4 (1992), pp. 284–294. DOI: 10.1002/(SICI)1097-4571(199205)43:4<284::AID-ASI3>3.0.CO;2-0.

[2] M. B. Clowes and J. R. Parks. "A New Technique in Automatic Character Recognition". In: *The Computer Journal* 4.2 (1961), pp. 121–128. DOI: 10.1093/comjnl/4.2.121.

[3] Yejing Xie. "Handwritten Mathematical Expression Recognition with Graph Neural Networks and Tree-based Language Models". Theses. Nantes Université, 2025. DOI: 10.70675/9d6a4ad5ze086z43d6z98dfz927ec012e817.

[4] Athenaeus of Naucratis. *The Deipnosophists, Book XII*. Trans. by Charles Duke Yonge. Paragraph 20. Online transcription of the 1854 translation. 1854. URL: https://www.attalus.org/old/athenaeus12a.html.

[5] Dorothea Blostein and Ann GRBAVEC. "Recognition of Mathematical Notation". In: *Handbook of Character Recognition and Document Image Analysis*, pp. 557–582. DOI: 10.1142/9789812830968_0021.

[6] Kam-Fai Chan and Dit-Yan Yeung. "Mathematical expression recognition: a survey". In: *International Journal on Document Analysis and Recognition* 3.1 (2000), pp. 3–15. ISSN: 1433-2833. DOI: 10.1007/PL00013549.

[7] Robert H. Anderson. "Syntax-directed recognition of hand-printed two-dimensional mathematics". In: *Symposium on Interactive Systems for Experimental Applied Mathematics: Proceedings of the Association for Computing Machinery Inc. Symposium*. Washington, D.C.: Association for Computing Machinery, 1967, pp. 436–459. ISBN: 9781450373098. DOI: 10.1145/2402536.2402585.

[8] Gabriel F. Groner. "Real-time recognition of handprinted text". In: *Proceedings of the November 7-10, 1966, fall joint computer conference on XX - AFIPS '66 (Fall)*. San Francisco, California: ACM Press, 1966, p. 591. DOI: 10.1145/1464291.1464355.

[9] Jaekyu Ha, R.M. Haralick, and I.T. Phillips. "Understanding mathematical expressions from document images". In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. Vol. 2. Montreal, Que., Canada: IEEE Comput. Soc. Press, 1995, pp. 956–959. ISBN: 978-0-8186-7128-9. DOI: 10.1109/ICDAR.1995.602060.

[10] H.M. Twaakyondo and M. Okamoto. "Structure analysis and recognition of mathematical expressions". In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. Vol. 1. Montreal, Que., Canada: IEEE Comput. Soc. Press, 1995, pp. 430–437. ISBN: 978-0-8186-7128-9. DOI: 10.1109/ICDAR.1995.599029.

[11] Y. Eto and M. Suzuki. "Mathematical formula recognition using virtual link network". In: *Proceedings of Sixth International Conference on Document Analysis and Recognition*. Seattle, WA, USA: IEEE Comput. Soc, 2001, pp. 762–767. ISBN: 978-0-7695-1263-1. DOI: 10.1109/ICDAR.2001.953891.

[12] Xiaode Zhang et al. "A Symbol Dominance Based Formulae Recognition Approach for PDF Documents". In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Vol. 01. 2017, pp. 1144–1149. DOI: 10.1109/ICDAR.2017.189.

[13] Masakazu Suzuki et al. "INFTY: an integrated OCR system for mathematical documents". In: *Proceedings of the 2003 ACM symposium on Document engineering - DocEng '03*. Grenoble, France: ACM Press, 2003, p. 95. ISBN: 978-1-58113-724-8. DOI: 10.1145/958220.958239.

[14] Masakazu Suzuki et al. "An Integrated OCR Software for Mathematical Documents and Its Output with Accessibility". In: *Computers Helping People with Special Needs*. Ed. by Takeo Kanade et al. Vol. 3118. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 648–655. ISBN: 978-3-540-22334-4 978-3-540-27817-7. DOI: 10.1007/978-3-540-27817-7_97.

[15] Christopher Malon, Seiichi Uchida, and Masakazu Suzuki. "Mathematical symbol recognition with support vector machines". In: *Pattern Recognition Letters* 29.9 (2008), pp. 1326–1332. ISSN: 0167-8655. DOI: 10.1016/j.patrec.2008.02.005.

[16] Corinna Cortes and Vladimir Vapnik. "Support-vector networks". In: *Machine Learning* 20.3 (1995), pp. 273–297. ISSN: 1573-0565. DOI: 10.1007/BF00994018.

[17] Fumitaka Kimura et al. "Improvement of handwritten Japanese character recognition using weighted direction code histogram". In: *Pattern Recognition* 30.8 (1997), pp. 1329–1337. ISSN: 00313203. DOI: 10.1016/S0031-3203(96)00153-7.

[18] Francisco Alvaro and Joan Andreu Sanchez. "Comparing Several Techniques for Offline Recognition of Printed Mathematical Symbols". In: *2010 20th International Conference on Pattern Recognition*. Istanbul, Turkey: IEEE, 2010, pp. 1953–1956. ISBN: 978-1-4244-7542-1. DOI: 10.1109/ICPR.2010.481.

[19] A.H. Toselli, A. Juan, and E. Vidal. "Spontaneous handwriting recognition and classification". In: *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* Cambridge, UK: IEEE, 2004, 433–436 Vol.1. ISBN: 978-0-7695-2128-2. DOI: 10.1109/ICPR.2004.1334151.

[20] K. Ashida et al. "Performance Evaluation of a Mathematical Formula Recognition System with a large scale of printed formula images". In: *Second International Conference on Document Image Analysis for Libraries (DIAL'06)*. Lyon, France: IEEE, 2006, pp. 320–331. ISBN: 978-0-7695-2531-0. DOI: 10.1109/DIAL.2006.30.

[21] Thanh-Nghia Truong et al. "A survey on handwritten mathematical expression recognition: The rise of encoder-decoder and GNN models". In: *Pattern Recogn.* 153.C (2024). ISSN: 0031-3203. DOI: 10.1016/j.patcog.2024.110531.

[22] Yuntian Deng et al. *Image-to-Markup Generation with Coarse-to-Fine Attention*. 2017. DOI: 10.48550/arXiv.1609.04938.

[23] Andrej Karpathy and Li Fei-Fei. *Deep Visual-Semantic Alignments for Generating Image Descriptions*. 2015. DOI: 10.48550/arXiv.1412.2306.

[24] Kelvin Xu et al. *Show, Attend and Tell: Neural Image Caption Generation with Visual Attention*. 2016. URL: https://arxiv.org/abs/1502.03044.

[25] Kishore Papineni et al. "BLEU: a method for automatic evaluation of machine translation". In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02*. Philadelphia, Pennsylvania: Association for Computational Linguistics, 2001, p. 311. DOI: 10.3115/1073083.1073135.

[26] Guillaume Genthial and Romain Sauvestre. *Image to LaTeX*. Stanford University CS231n Course Report. 2017. URL: https://cs231n.stanford.edu/reports/2017/pdfs/815.pdf.

[27] Sumeet S. Singh. *Teaching Machines to Code: Neural Markup Generation with Visual Attention*. 2018. URL: https://arxiv.org/abs/1802.05415.

[28] Ghaith Bilbeisi, Sheraz Ahmed, and Rupak Majumdar. "DeepEquaL: Deep Learning Based Mathematical Equation to Latex Generation". In: *Neural Information Processing*. Ed. by Haiqin Yang et al. Vol. 1333. Cham: Springer International Publishing, 2020, pp. 324–332. ISBN: 978-3-030-63822-1 978-3-030-63823-8. DOI: 10.1007/978-3-030-63823-8_38.

[29] Yu Yin et al. "Transcribing Content from Structural Images with Spotlight Mechanism". In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2018, pp. 2643–2652. DOI: 10.1145/3219819.3219962.

[30] Jian Wang, Yunchuan Sun, and Shenling Wang. "Image To Latex with DenseNet Encoder and Joint Attention". In: *Procedia Computer Science* 147 (2019), pp. 374–380. ISSN: 18770509. DOI: 10.1016/j.procs.2019.01.246.

[31] Wei Zhang, Zhiqiang Bai, and Yuesheng Zhu. "An Improved Approach Based on CNN-RNNs for Mathematical Expression Recognition". In: *Proceedings of the 2019 4th International Conference on Multimedia Systems and Signal Processing*. Guangzhou China: ACM, 2019, pp. 57–61. ISBN: 978-1-4503-7171-1. DOI: 10.1145/3330393.3330410.

[32] Zelun Wang and Jyh-Charn Liu. "Translating math formula images to LaTeX sequences using deep neural networks with sequence-level training". In: *International Journal on Document Analysis and Recognition (IJDAR)* 24.1-2 (2021), pp. 63–75. ISSN: 1433-2833, 1433-2825. DOI: 10.1007/s10032-020-00360-2.

[33] Abolfazl Mirkazemy et al. "Mathematical expression recognition using a new deep neural model". In: *Neural Networks* 167 (2023), pp. 865–874. ISSN: 08936080. DOI: 10.1016/j.neunet.2023.08.045.

[34] Zuoyu Yan et al. *ConvMath: A Convolutional Sequence Network for Mathematical Expression Recognition*. 2020. URL: https://arxiv.org/abs/2012.12619.

[35] Dan Anitei, Joan Andreu Sánchez, and José Miguel Benedí. "Py4MER: A CTC-Based Mathematical Expression Recognition System". In: *Pattern Recognition and Image Analysis*. Ed. by Antonio Pertusa et al. Vol. 14062. Cham: Springer Nature Switzerland, 2023, pp. 426–441. ISBN: 978-3-031-36615-4 978-3-031-36616-1. DOI: 10.1007/978-3-031-36616-1_34.

[36] Razvan Pascanu et al. *How to Construct Deep Recurrent Neural Networks*. 2014. DOI: 10.48550/arXiv.1312.6026.

[37] Richard Zhang. *Making Convolutional Networks Shift-Invariant Again*. 2019. DOI: 10.48550/arXiv.1904.11486.

[38] Jonas Gehring et al. *Convolutional Sequence to Sequence Learning*. 2017. URL: https://arxiv.org/abs/1705.03122.

[39] Ashish Vaswani et al. *Attention Is All You Need*. 2017. URL: https://arxiv.org/abs/1706.03762.

[40] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. URL: https://arxiv.org/abs/2010.11929.

[41] Ze Liu et al. *Swin Transformer: Hierarchical Vision Transformer using Shifted Windows*. 2021. URL: https://arxiv.org/abs/2103.14030.

[42] Haiping Wu et al. *CvT: Introducing Convolutions to Vision Transformers*. 2021. DOI: 10.48550/arXiv.2103.15808.

[43] Kenton Lee et al. "Pix2Struct: screenshot parsing as pretraining for visual language understanding". In: *Proceedings of the 40th International Conference on Machine Learning*. ICML'23. Honolulu, Hawaii, USA: JMLR.org, 2023. DOI: 10.48550/arXiv.2210.03347.

[44] Geewook Kim et al. *OCR-free Document Understanding Transformer*. 2022. URL: https://arxiv.org/abs/2111.15664.

[45] Minghao Li et al. *TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models*. 2022. DOI: 10.48550/arXiv.2109.10282.

[46] Lukas Blecher et al. *Nougat: Neural Optical Understanding for Academic Documents*. 2023. URL: https://arxiv.org/abs/2308.13418.

[47] Lukas Blecher. *LaTeX-OCR*. Accessed: 2026-07-13. 2022. URL: https://github.com/lukas-blecher/LaTeX-OCR.

[48] Anh Duy Le et al. "A Hybrid Vision Transformer Approach for Mathematical Expression Recognition". In: *2022 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. Sydney, Australia: IEEE, 2022, pp. 1–7. ISBN: 978-1-6654-5642-5. DOI: 10.1109/DICTA56598.2022.10034626.

[49] Van-Xung Vu et al. "Transformer-based method for mathematical expression recognition in document images". In: *2022 International Conference on Multimedia Analysis and Pattern Recognition (MAPR)*. 2022, pp. 1–6. DOI: 10.1109/MAPR56351.2022.9924740.

[50] Dan Anitei et al. "Improving Efficiency and Performance Through CTC-Based Transformers for Mathematical Expression Recognition". In: *Document Analysis and Recognition - ICDAR*. Ed. by Elisa H. Barney Smith, Marcus Liwicki, and Liangrui Peng. Vol. 14808. Cham: Springer Nature Switzerland, 2024, pp. 3–20. ISBN: 978-3-031-70548-9 978-3-031-70549-6. DOI: 10.1007/978-3-031-70549-6_1.

[51] Shuaijian Ji et al. "Imbalanced Conditional Conv-Transformer for Mathematical Expression Recognition". In: *Artificial Neural Networks and Machine Learning – ICANN 2023*. Ed. by Lazaros Iliadis et al. Vol. 14259. Cham: Springer Nature Switzerland, 2023, pp. 446–458. ISBN: 978-3-031-44222-3 978-3-031-44223-0. DOI: 10.1007/978-3-031-44223-0_36.

[52] Yingnan Fu et al. *EDSL: An Encoder-Decoder Architecture with Symbol-Level Features for Printed Mathematical Expression Recognition*. 2020. URL: https://arxiv.org/abs/2007.02517.

[53] Yuqing Wang et al. *Dual Branch Network Towards Accurate Printed Mathematical Expression Recognition*. 2023. URL: https://arxiv.org/abs/2312.09030.

[54] Zhaokun Zhou et al. "TRMER: Transformer-Based End to End Printed Mathematical Expression Recognition". In: *2023 International Joint Conference on Neural Networks (IJCNN)*. Gold Coast, Australia: IEEE, 2023, pp. 1–6. ISBN: 978-1-6654-8867-9. DOI: 10.1109/IJCNN54540.2023.10191139.

[55] Jiale Wang et al. *Enhancing Complex Formula Recognition with Hierarchical Detail-Focused Network*. 2025. DOI: 10.48550/arXiv.2409.11677.

[56] Vik Paruchuri. *Texify*. Accessed: 2026-07-13. 2024. URL: https://github.com/VikParuchuri/texify.

[57] Haoyang Li et al. *Towards Scalable Training for Handwritten Mathematical Expression Recognition*. 2026. DOI: 10.48550/arXiv.2508.09220.

[58] Felix M. Schmitt-Koopmann et al. "MathNet: A Data-Centric Approach for Printed Mathematical Expression Recognition". In: *IEEE Access* 12 (2024). Conference Name: IEEE Access, pp. 76963–76974. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2024.3404834.

[59] Bin Wang et al. *UniMERNet: A Universal Network for Real-World Mathematical Expression Recognition*. 2024. DOI: 10.48550/arXiv.2404.15254.

[60] Yejing Xie et al. "ICDAR 2023 CROHME: Competition on Recognition of Handwritten Mathematical Expressions". In: *Document Analysis and Recognition - ICDAR*. Ed. by Gernot A. Fink et al. Vol. 14188. Cham: Springer Nature Switzerland, 2023, pp. 553–565. ISBN: 978-3-031-41678-1 978-3-031-41679-8. DOI: 10.1007/978-3-031-41679-8_33.

[61] Xiaoyan Lin et al. "Performance Evaluation of Mathematical Formula Identification". In: *2012 10th IAPR International Workshop on Document Analysis Systems*. 2012, pp. 287–291. DOI: 10.1109/DAS.2012.68.

[62] Bui Hai Phong et al. "A deep learning based system for mathematical expression detection and recognition in document images". In: *2020 12th International Conference on Knowledge and Systems Engineering (KSE)*. 2020, pp. 85–90. DOI: 10.1109/KSE50997.2020.9287693.

[63] Asher Trockman and J. Zico Kolter. *Patches Are All You Need?* 2022. DOI: 10.48550/arXiv.2201.09792.

[64] Jianwei Yang et al. "Focal Attention for Long-Range Interactions in Vision Transformers". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 30008–30022. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/fc1a36821b02abbd2503fd949bfc9131-Paper.pdf.

[65] Yinhan Liu et al. *Multilingual Denoising Pre-training for Neural Machine Translation*. 2020. DOI: 10.48550/arXiv.2001.08210.

[66] Vik Paruchuri. *Surya*. Accessed: 2026-07-13. 2024. URL: https://github.com/datalab-to/surya.

[67] OpenAI. *GPT-4V(ision) System Card*. Tech. rep. Accessed: 2026-07-30. OpenAI, 2023. URL: https://cdn.openai.com/papers/GPTV_System_Card.pdf.

[68] Shuai Bai et al. *Qwen3-VL Technical Report*. 2025. arXiv: 2511.21631 [cs.CV]. URL: https://arxiv.org/abs/2511.21631.

[69] Pan Lu et al. *MathVista: Evaluating Mathematical Reasoning of Foundation Models in Visual Contexts*. 2024. DOI: 10.48550/arXiv.2310.02255.

[70] Renrui Zhang et al. *MathVerse: Does Your Multi-modal LLM Truly See the Diagrams in Visual Math Problems?* 2024. DOI: 10.48550/arXiv.2403.14624.

[71] Merouane Zouaid, Yejing Xie, and Harold Mouchère. "Boosting Handwritten Mathematical Expression Recognition Through Contextual Reasoning with Vision Large Language Models (vLLMs)". In: *Document Analysis and Recognition – ICDAR*. Wuhan, China: Springer-Verlag, 2025, pp. 313–326. ISBN: 978-3-032-09367-7. DOI: 10.1007/978-3-032-09368-4_19.

[72] Weikang Bai et al. *Complex Mathematical Expression Recognition: Benchmark, Large-Scale Dataset and Strong Baseline*. 2025. DOI: 10.48550/arXiv.2512.13731.

[73] Xin Zhou et al. *LessLeak-Bench: A First Investigation of Data Leakage in LLMs Across 83 Software Engineering Benchmarks*. 2025. DOI: 10.48550/arXiv.2502.06215.

[74] Hao Feng et al. *Dolphin: Document Image Parsing via Heterogeneous Anchor Prompting*. 2025. DOI: 10.48550/arXiv.2505.14059.

[75] Zhang Li et al. *MonkeyOCR: Document Parsing with a Structure-Recognition-Relation Triplet Paradigm*. 2026. DOI: 10.48550/arXiv.2506.05218.

[76] Yumeng Li et al. *dots.ocr: Multilingual Document Layout Parsing in a Single Vision-Language Model*. 2025. DOI: 10.48550/arXiv.2512.02498.

[77] Junbo Niu et al. *MinerU2.5: A Decoupled Vision-Language Model for Efficient High-Resolution Document Parsing*. 2025. DOI: 10.48550/arXiv.2509.22186.

[78] Cheng Cui et al. *PaddleOCR-VL: Boosting Multilingual Document Parsing via a 0.9B Ultra-Compact Vision-Language Model*. 2025. DOI: 10.48550/arXiv.2510.14528.

[79] Cheng Cui et al. *PaddleOCR-VL-1.5: Towards a Multi-Task 0.9B VLM for Robust In-the-Wild Document Parsing*. 2026. DOI: 10.48550/arXiv.2601.21957.

[80] Haoran Wei, Yaofeng Sun, and Yukun Li. *DeepSeek-OCR: Contexts Optical Compression*. 2025. arXiv: 2510.18234 [cs.CV]. URL: https://arxiv.org/abs/2510.18234.

[81] Haoran Wei, Yaofeng Sun, and Yukun Li. *DeepSeek-OCR 2: Visual Causal Flow*. 2026. DOI: 10.48550/arXiv.2601.20552.

[82] Linke Ouyang et al. *OmniDocBench: Benchmarking Diverse PDF Document Parsing with Comprehensive Annotations*. 2025. DOI: 10.48550/arXiv.2412.07626.

[83] Harold Mouchere et al. "CROHME2011: Competition on Recognition of Online Handwritten Mathematical Expressions". In: *2011 International Conference on Document Analysis and Recognition*. Beijing, China: IEEE, 2011, pp. 1497–1500. ISBN: 978-1-4577-1350-7. DOI: 10.1109/ICDAR.2011.297.

[84] Richard Zanibbi, Harold Mouchère, and Christian Viard-Gaudin. "Evaluating structural pattern recognition for handwritten math via primitive label graphs". In: *Document Recognition and Retrieval XX*. Ed. by Richard Zanibbi and Bertrand Coüasnon. Vol. 8658. International Society for Optics and Photonics. SPIE, 2013, p. 865817. DOI: 10.1117/12.2008409.

[85] Philippe Gervais, Anastasiia Fadeeva, and Andrii Maksai. *MathWriting: A Dataset For Handwritten Mathematical Expression Recognition*. 2025. DOI: 10.48550/arXiv.2404.10690.

[86] Theodore William Chaundy and Oxford University Press. *The printing of mathematics : aids for authors and editors and rules for compositors and readers at the University Press, Oxford / by T.W. Chaundy, P.R. Barrett and Charles Batey*. Rev. London: Oxford University Press, 1957.

[87] F. Cajori. *A History of Mathematical Notations*. A History of Mathematical Notations 1–2 tom. Dover Publications, 1993. ISBN: 9780486677668. URL: https://books.google.fr/books?id=7juWmvQSTvwC.

[88] Leslie Lamport. *LATEX: a document preparation system*. OCLC: 12550262. Reading, Mass.: Addison-Wesley Pub. Co., 1986. ISBN: 978-0-201-15790-1.

[89] Robert Miner. “The Importance of MathML to Mathematics Communication”. In: *Notices of the American Mathematical Society* 52 (2005). Accessed: 2026-07-13. URL: https://www.ams.org/journals/notices/200505/fea-miner.pdf.

[90] T.W. Cole. “MathML in practice: issues and promise”. In: *Data Science Journal* 5 (2006), pp. 209–218. ISSN: 1683-1470. DOI: 10.2481/dsj.5.209.

[91] André Greiner-Petter et al. “MathTools: An Open API for Convenient MathML Handling”. In: *Intelligent Computer Mathematics*. Ed. by Florian Rabe et al. Vol. 11006. Cham: Springer International Publishing, 2018, pp. 104–110. ISBN: 978-3-319-96811-7 978-3-319-96812-4. DOI: 10.1007/978-3-319-96812-4_9.

[92] Bruce R. Miller. “Three Years of DLMF: Web, Math and Search”. In: *Intelligent Computer Mathematics*. Ed. by Jacques Carette et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 288–295. ISBN: 978-3-642-39320-4.

[93] Abdou Youssef. *math-dlmf-dataset*. Accessed: 2026-07-30. 2020. URL: https://github.com/abdouyoussef/math-dlmf-dataset.

[94] Mike Dewar. “OpenMath: an overview”. In: *SIGSAM Bull.* 34.2 (2000), pp. 2–5. ISSN: 0163-5824. DOI: 10.1145/362001.362008.

[95] Richard J. Fateman. *A Critique of OpenMath and Thoughts on Encoding Mathematics*. Technical Report. Accessed: 2026-07-30. Computer Science Division, University of California, Berkeley, Jan. 17, 2001. URL: https://people.eecs.berkeley.edu/~fateman/papers/openmathcrit.pdf.

[96] Christian Feichter and Tim Schlippe. “Investigating Models for the Transcription of Mathematical Formulas in Images”. In: *Applied Sciences* 14.3 (2024), p. 1140. ISSN: 2076-3417. DOI: 10.3390/app14031140.

[97] Kyudan Jung et al. *MathBridge: A Large Corpus Dataset for Translating Spoken Mathematical Expressions into LaTeX Formulas for Improved Readability*. 2024. arXiv: 2408.07081 [cs.LG]. URL: https://arxiv.org/abs/2408.07081.

[98] Solen Quiniou et al. “HAMEX - A Handwritten and Audio Dataset of Mathematical Expressions”. In: *2011 International Conference on Document Analysis and Recognition*. Beijing, China: IEEE, 2011, pp. 452–456. ISBN: 978-1-4577-1350-7. DOI: 10.1109/ICDAR.2011.97.

[99] Mahshad Mahdavi and Richard Zanibbi. “Tree-Based Structure Recognition Evaluation for Math Expressions: Techniques and Case Study”. In: *Proceedings of the 13th IAPR International Workshop on Graphics Recognition (GREC 2019)*. Extended abstract, 2 pages. Workshop held in conjunction with ICDAR 2019. Sydney, Australia, 2019. URL: https://www.cs.rit.edu/~rlaz/files/Mahdavi_GREC2019.pdf.

[100] S. Uchida, A. Nomura, and M. Suzuki. "Quantitative analysis of mathematical documents". In: *International Journal of Document Analysis and Recognition (IJDAR)* 7.4 (2005), pp. 211–218. ISSN: 1433-2833, 1433-2825. DOI: 10.1007/s10032-005-0142-y.

[101] Ihsin T. Phillips. "Methodologies for using UW databases for OCR and image-understanding systems". In: *Document Recognition V*. Vol. 3305. SPIE, 1998, p. 112. DOI: 10.1117/12.304624.

[102] M. Suzuki, S. Uchida, and A. Nomura. "A ground-truthed mathematical character and symbol image database". In: *Eighth International Conference on Document Analysis and Recognition (ICDAR)*. Seoul, South Korea: IEEE, 2005, 675–679 Vol. 2. ISBN: 978-0-7695-2420-7. DOI: 10.1109/ICDAR.2005.14.

[103] R. Vente. *im2latex170k*. Accessed: 2026-07-13. 2023. URL: https://www.kaggle.com/datasets/rvente/im2latex170k/data.

[104] Gregory Eritsyan. *IM2LATEX-230k*. Accessed: 2026-07-13. 2023. URL: https://www.kaggle.com/datasets/gregoryeritsyan/im2latex-230k/data.

[105] Dan Anitei et al. "The IBEM dataset: A large printed scientific image dataset for indexing and searching mathematical expressions". In: *Pattern Recognition Letters* 172 (2023), pp. 29–36. ISSN: 01678655. DOI: 10.1016/j.patrec.2023.05.033.

[106] Quoc Trung Hoang. *Fusion Image-to-LaTeX Datasets*. Accessed: 2026-07-13. 2024. URL: https://huggingface.co/datasets/hoang-quoc-trung/fusion-image-to-latex-datasets.

[107] Johannes Gehrke, Paul Ginsparg, and Jon Kleinberg. "Overview of the 2003 KDD Cup". In: *ACM SIGKDD Explorations Newsletter* 5.2 (2003), pp. 149–151. ISSN: 1931-0145, 1931-0153. DOI: 10.1145/980972.980992.

[108] Gregory Marus. *Printed-LaTeX-Data-Generation*. Accessed: 2026-07-13. 2022. URL: https://github.com/gmarus777/Printed-Latex-Data-Generation.

[109] Pearson. *Aida Calculus Math Handwriting Recognition Dataset*. Accessed: 2026-07-13. 2020. URL: https://www.kaggle.com/datasets/aidapearson/ocr-data.

[110] Tianrui Zong et al. "ICPR 2024 Competition on Multi-line Mathematical Expressions Recognition". In: *Pattern Recognition. Competitions*. Ed. by Apostolos Antonacopoulos et al. Vol. 15334. Cham: Springer Nature Switzerland, 2025, pp. 17–29. ISBN: 978-3-031-80138-9 978-3-031-80139-6. DOI: 10.1007/978-3-031-80139-6_2.

[111] Minesh Mathew, Dimosthenis Karatzas, and C. V. Jawahar. "DocVQA: A Dataset for VQA on Document Images". In: *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. Waikoloa, HI, USA: IEEE, 2021, pp. 2199–2208. ISBN: 978-1-6654-0477-8. DOI: 10.1109/WACV48630.2021.00225.

[112] Jake Poznanski et al. *olmOCR: Unlocking Trillions of Tokens in PDFs with Vision Language Models*. 2025. DOI: 10.48550/arXiv.2502.18443.

[113] Ling Fu et al. *OCRBench v2: An Improved Benchmark for Evaluating Large Multimodal Models on Visual Text Localization and Reasoning*. 2025. DOI: 10.48550/arXiv.2501.00321.

[114] Zhibo Yang et al. *CC-OCR: A Comprehensive and Challenging OCR Benchmark for Evaluating Large Multimodal Models in Literacy*. 2024. DOI: 10.48550/arXiv.2412.02210.

[115] An-Lan Wang et al. *WildDoc: How Far Are We from Achieving Comprehensive and Robust Document Understanding in the Wild?* 2025. DOI: 10.48550/arXiv.2505.11015.

[116] Yongkun Du et al. *DocPTBench: Benchmarking End-to-End Photographed Document Parsing and Translation*. 2025. DOI: 10.48550/arXiv.2511.18434.

[117] Changda Zhou et al. *Real5-OmniDocBench: A Full-Scale Physical Reconstruction Benchmark for Robust Document Parsing in the Wild*. 2026. DOI: 10.48550/arXiv.2603.04205.

[118] Michal Kucer et al. "DeepPatent: Large scale patent drawing recognition and retrieval". In: *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. Waikoloa, HI, USA: IEEE, 2022, pp. 557–566. ISBN: 978-1-6654-0915-5. DOI: 10.1109/WACV51458.2022.00063.

[119] Kehinde Ajayi et al. "DeepPatent2: A Large-Scale Benchmarking Corpus for Technical Drawing Understanding". In: *Scientific Data* 10.1 (2023), p. 772. ISSN: 2052-4463. DOI: 10.1038/s41597-023-02653-7.

[120] Shreya Shukla et al. *PatentLMM: Large Multimodal Model for Generating Descriptions for Patent Figures*. 2025. DOI: 10.48550/arXiv.2501.15074.

[121] Xin Wei et al. "Visual descriptor extraction from patent figure captions: a case study of data efficiency between BiLSTM and transformer". In: *Proceedings of the 22nd ACM/IEEE Joint Conference on Digital Libraries*. Cologne Germany: ACM, 2022, pp. 1–5. ISBN: 978-1-4503-9345-4. DOI: 10.1145/3529372.3533299.

[122] Mirac Suzgun et al. *The Harvard USPTO Patent Dataset: A Large-Scale, Well-Structured, and Multi-Purpose Corpus of Patent Applications*. 2022. URL: https://arxiv.org/abs/2207.04043.

[123] Eva Sharma, Chen Li, and Lu Wang. "BIGPATENT: A Large-Scale Dataset for Abstractive and Coherent Summarization". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, 2019, pp. 2204–2213. DOI: 10.18653/v1/P19-1212.

[124] Stefanos Vrochidis et al. "Towards content-based patent image retrieval: A framework perspective". In: *World Patent Information* 32.2 (2010), pp. 94–106. ISSN: 01722190. DOI: 10.1016/j.wpi.2009.05.010.

[125] Julian Risch et al. *PatentMatch: A Dataset for Matching Patent Claims & Prior Art*. 2020. DOI: 10.48550/arXiv.2012.13919.

[126] Lekang Jiang, Chengzu Li, and Stefan Goetz. "Enriching Patent Claim Generation with European Patent Dataset". In: *Findings of the Association for Computational Linguistics: EMNLP 2025*. Ed. by Christos Christodoulopoulos et al. Suzhou, China: Association for Computational Linguistics, 2025, pp. 7734–7751. ISBN: 979-8-89176-335-7. DOI: 10.18653/v1/2025.findings-emnlp.408.

[127] Dana Aubakirova, Kim Gerdes, and Lufei Liu. "PatFig: Generating Short and Long Captions for Patent Figures". In: *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*. Paris, France: IEEE, 2023, pp. 2835–2841. ISBN: 979-8-3503-0744-3. DOI: 10.1109/ICCVW60793.2023.00305.

[128] Kenneth Heafield et al. "The EuroPat Corpus: A Parallel Corpus of European Patent Data". In: *Proceedings of the Thirteenth Language Resources and Evaluation Conference*. Ed. by Nicoletta Calzolari et al. Marseille, France: European Language Resources Association, 2022, pp. 732–740. URL: https://aclanthology.org/2022.lrec-1.78/.

[129] Florina Piroi et al. "CLEF-IP 2011: Retrieval in the Intellectual Property Domain". In: *Conference and Labs of the Evaluation Forum*. 2011. URL: https://api.semanticscholar.org/CorpusID:12711052.

[130] Adrien Lapointe and Dorothea Blostein. "Issues in Performance Evaluation: A Case Study of Math Recognition". In: *2009 10th International Conference on Document Analysis and Recognition*. Barcelona, Spain: IEEE, 2009, pp. 1355–1359. ISBN: 978-1-4244-4500-4. DOI: 10.1109/ICDAR.2009.247.

[131] Ahmad-Montaser Awal, Harold Mouchere, and Christian Viard-Gaudin. "The Problem of Handwritten Mathematical Expression Recognition Evaluation". In: *2010 12th International Conference on Frontiers in Handwriting Recognition*. Kolkata, India: IEEE, 2010, pp. 646–651. ISBN: 978-1-4244-8353-2. DOI: 10.1109/ICFHR.2010.106.

[132] José Grimm. *Converting LaTeX to MathML: the Tralics algorithms*. Research Report RR-6373. INRIA, 2007, p. 16. URL: https://inria.hal.science/inria-00192610.

[133] George van Asch. *BlahtexML*. 2006. URL: https://github.com/gvanas/blahtexml.

[134] Bruce R. Miller. *LaTeXML*. 2004. URL: https://github.com/brucemiller/LaTeXML.

[135] Heinrich Stamerjohanns et al. "Transforming Large Collections of Scientific Publications to XML". In: *Mathematics in Computer Science* 3.3 (2010), pp. 299–307. DOI: 10.1007/s11786-010-0024-7.

[136] Heinrich Stamerjohanns et al. "MathML-aware Article Conversion from LaTeX". In: *Towards a Digital Mathematics Library*. Masaryk University Press, 2009, pp. 109–120. URL: https://eudml.org/doc/220017.

[137] Yuntian Deng, Noriyuki Kojima, and Alexander M. Rush. *Markup-to-Image Diffusion Models with Scheduled Sampling*. 2022. arXiv: 2210.05147 [cs.LG]. URL: https://arxiv.org/abs/2210.05147.

[138] David Formanek et al. "Normalization of Digital Mathematics Library Content". In: *Proceedings of the 5th Workshop Towards a Digital Mathematics Library*. Vol. 921. CEUR Workshop Proceedings. CEUR-WS.org, 2012, pp. 91–103. URL: https://ceur-ws.org/Vol-921/wip-05.pdf.

[139] Mohammed Shatnawi and Abdou Youssef. "Equivalence Detection Using Parse-tree Normalization for Math Search". In: *2007 2nd International Conference on Digital Information Management*. Lyon, France: IEEE, 2007, pp. 643–648. DOI: 10.1109/ICDIM.2007.4444297.

[140] V. I. Levenshtein. "Binary Codes Capable of Correcting Deletions, Insertions, and Reversals". In: *Doklady Akademii Nauk SSSR* 163.4 (1965), pp. 845–848. URL: https://nymity.ch/sybilhunting/pdf/Levenshtein1966a.pdf.

[141] Alexis Roche et al. "The correlation ratio as a new similarity measure for multimodal image registration". In: *Medical Image Computing and Computer-Assisted Intervention — MICCAI'98*. Ed. by William M. Wells, Alan Colchester, and Scott Delp. Vol. 1496. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 1115–1124. ISBN: 978-3-540-65136-9 978-3-540-49563-5. DOI: 10.1007/BFb0056301.

[142] Zhou Wang et al. "Image Quality Assessment: From Error Visibility to Structural Similarity". In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612. DOI: 10.1109/TIP.2003.819861.

[143] Richard Zanibbi and Bo Yuan. "Keyword and Image-Based Retrieval for Mathematical Expressions". In: *Document Recognition and Retrieval XVIII*. Vol. 7874. Proceedings of SPIE. San Francisco, CA, USA: SPIE, 2011, p. 78740I. DOI: 10.1117/12.873312.

[144] Francisco Alvaro, Joan-Andreu Sanchez, and Jose-Miguel Benedı. "IMEGE: Image-based Mathematical Expression Global Error". In: (2013).

[145] Bin Wang et al. *Image Over Text: Transforming Formula Recognition Evaluation with Character Detection Matching*. 2025. DOI: 10.48550/arXiv.2409.03643.

[146] Kunal Sain, Abhishek Dasgupta, and Utpal Garain. "EMERS: A Tree Matching-Based Performance Evaluation of Mathematical Expression Recognition Systems". In: *International Journal on Document Analysis and Recognition* 14.1 (2011), pp. 75–85. DOI: 10.1007/s10032-010-0121-9.

[147] Keisuke Yokoi and Akiko Aizawa. "An Approach to Similarity Search for Mathematical Expressions Using MathML". In: *Towards a Digital Mathematics Library*. Brno: Masaryk University Press, 2009, pp. 27–35. URL: https://eudml.org/doc/221460.

[148] Yuping Qin, Junnan Guo, and Aihua Zhang. "A Mathematical Formula Matching Algorithm Based on MathML". In: *Proceedings of the International Conference on Logistics, Engineering, Management and Computer Science*. Atlantis Press, 2015, pp. 1681–1684. ISBN: 978-94-6252-102-5. DOI: 10.2991/lemcs-15.2015.339.

[149] Tianyi Zhang et al. *BERTScore: Evaluating Text Generation with BERT*. 2020. DOI: 10.48550/arXiv.1904.09675.

[150] Ricardo Rei et al. *COMET: A Neural Framework for MT Evaluation*. 2020. URL: https://arxiv.org/abs/2009.09025.

[151] Shuai Peng et al. *MathBERT: A Pre-Trained Model for Mathematical Formula Understanding*. 2021. DOI: 10.48550/arXiv.2105.00377.

[152] Kyudan Jung et al. *TeXBLEU: Automatic Metric for Evaluate LaTeX Format*. 2024. DOI: 10.48550/arXiv.2409.06639.

[153] Liangcai Gao et al. *Preliminary Exploration of Formula Embedding for Mathematical Information Retrieval: can mathematical formulae be embedded like a natural language?* 2017. URL: https://arxiv.org/abs/1707.05154.

[154] Kriste Krstovski and David M. Blei. *Equation Embeddings*. 2018. DOI: 10.48550/arXiv.1803.09123.

[155] Philipp Scharpf et al. *Discovery and Recognition of Formula Concepts using Machine Learning*. 2023. DOI: 10.48550/arXiv.2303.01994.

[156] Lukas Pfahler, Jonathan Schill, and Katharina Morik. "The Search for Equations – Learning to Identify Similarities Between Mathematical Expressions". In: *Machine Learning and Knowledge Discovery in Databases*. Springer International Publishing, 2020, pp. 704–718. ISBN: 9783030461324. DOI: 10.1007/978-3-030-46133-1_42.

[157] Zichao Wang et al. "Scientific Formula Retrieval via Tree Embeddings". In: *2021 IEEE International Conference on Big Data (Big Data)*. 2021, pp. 1493–1503. DOI: 10.1109/BigData52589.2021.9671942.

[158] Saleem Ahmed et al. "Equation Attention Relationship Network (EARN) : A Geometric Deep Metric Framework for Learning Similar Math Expression Embedding". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. Milan, Italy: IEEE, 2021, pp. 6282–6289. ISBN: 978-1-7281-8808-9. DOI: 10.1109/ICPR48806.2021.9412619.

[159] François Wieckowiak et al. "A Multimodal Evaluation Pipeline for Mathematical Expression Recognition: Comparisons of Datasets, Metrics, and Models". In: *Document Analysis and Recognition – ICDAR*. 2025, pp. 120–136. ISBN: 978-3-032-04627-7. DOI: 10.1007/978-3-032-04627-7_7.

[160] Sumeet S. Singh and Sergey Karayev. "Full Page Handwriting Recognition via Image to Sequence Extraction". In: *Document Analysis and Recognition – ICDAR*. 2021, pp. 55–69. ISBN: 978-3-030-86334-0. DOI: 10.1007/978-3-030-86334-0_4.

[161] Xu Zhong, Elaheh ShafieiBavani, and Antonio Jimeno Yepes. *Image-based table recognition: data, model, and evaluation*. 2020. arXiv: 1911.10683 [cs.CV]. URL: https://arxiv.org/abs/1911.10683.

[162] Brandon Smock, Rohith Pesala, and Robin Abraham. *GriTS: Grid table similarity metric for table structure recognition*. 2023. arXiv: 2203.12555 [cs.LG]. URL: https://arxiv.org/abs/2203.12555.

[163] Dávid Bajusz, Anita Rácz, and Károly Héberger. "Why is Tanimoto index an appropriate choice for fingerprint-based similarity calculations?" In: *Journal of Cheminformatics* 7.1 (2015), p. 20. ISSN: 1758-2946. DOI: 10.1186/s13321-015-0069-3.

[164] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV]. URL: https://arxiv.org/abs/1512.03385.

[165] OleehyO. *OleehyO/TexTeller*. Accessed 2026-02-17. 2024. URL: https://huggingface.co/OleehyO/TexTeller.

[166] Lukas Blecher. *lukas-blecher/LaTeX-OCR*. Accessed 2026-02-17. 2021. URL: https://github.com/lukas-blecher/LaTeX-OCR.

[167] Prithiv Sakthi. *prithivMLmods/Qwen2-VL-OCR2-2B-Instruct*. Accessed 2026-02-17. 2025. URL: https://huggingface.co/prithivMLmods/Qwen2-VL-OCR2-2B-Instruct.

[168] Mostafa Dehghani et al. *Patch n' Pack: NaViT, a Vision Transformer for any Aspect Ratio and Resolution*. 2023. arXiv: 2307.06304 [cs.CV]. URL: https://arxiv.org/abs/2307.06304.

[169] François Wieckowiak et al. "PatentME: A Dataset and Reference-Free Post-OCR Verification Task for Printed Mathematical Expression Recognition". In: *Document Analysis and Recognition – ICDAR*. 2026, pp. 611–628. ISBN: 978-3-032-36023-6. DOI: 10.1007/978-3-032-36023-6_35.

[170] Richard Zanibbi, Behrooz Mansouri, and Anurag Agarwal. "Mathematical Information Retrieval: Search and Question Answering". In: *Foundations and Trends® in Information Retrieval* 19.1-2 (2025), pp. 1–190. ISSN: 1554-0669, 1554-0677. DOI: 10.1561/1500000095.

[171] Richard Zanibbi and Li Yu. "Math Spotting: Retrieving Math in Technical Documents Using Handwritten Query Images". In: *2011 International Conference on Document Analysis and Recognition*. 2011, pp. 446–451. DOI: 10.1109/ICDAR.2011.96.

[172] Kenny Davila et al. "Tangent-V: Math Formula Image Search Using Line-of-Sight Graphs". In: *Advances in Information Retrieval*. Ed. by Leif Azzopardi et al. Vol. 11437. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 681–695. DOI: 10.1007/978-3-030-15712-8_44.

[173] Richard Zanibbi et al. "Multi-Stage Math Formula Search: Using Appearance-Based Similarity Metrics at Scale". In: *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 2016, pp. 145–154. DOI: 10.1145/2911451.2911512.

[174] Kenny Davila and Richard Zanibbi. "Layout and Semantics: Combining Representations for Mathematical Formula Search". In: *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2017, pp. 1165–1168. DOI: 10.1145/3077136.3080748.

[175] Zichao Wang et al. "Scientific Formula Retrieval via Tree Embeddings". In: *2021 IEEE International Conference on Big Data (Big Data)*. 2021, pp. 1493–1503. DOI: 10.1109/BigData52589.2021.9671942.

[176] Richard Zanibbi, Harold Mouchère, and Christian Viard-Gaudin. "Evaluating structural pattern recognition for handwritten math via primitive label graphs". In: *SPIE Proceedings*. IS&T/SPIE Electronic Imaging. Vol. 8658. Burlingame, California, USA: SPIE, 2013, p. 865817. DOI: 10.1117/12.2008409.

[177] Francisco Álvaro, Joan-Andreu Sánchez, and José-Miguel Benedí. "An Image-Based Measure for Evaluation of Mathematical Expression Recognition". In: *Pattern Recognition and Image Analysis*. Ed. by João M. Sanches, Luisa Micó, and Jaime S. Cardoso. Springer, 2013, pp. 682–690. DOI: 10.1007/978-3-642-38628-2_81.

[178] Jane Bromley et al. "Signature verification using a "Siamese" time delay neural network". In: *Proceedings of the 7th International Conference on Neural Information Processing Systems*. NIPS'93. https://dl.acm.org/doi/10.5555/2987189.2987282. 1993, pp. 737–744.

[179] Berat Kurar Barakat, Reem Alasam, and Jihad El-Sana. "Word Spotting Using Convolutional Siamese Network". In: *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*. 2018, pp. 229–234. DOI: 10.1109/das.2018.67.

[180] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. "Siamese Neural Networks for One-shot Image Recognition". In: *Proceedings of the 32nd International Conference on Machine Learning*. https://www.cs.cmu.edu/~rsalakhu/papers/oneshot1.pdf. 2015.

[181] Asma Naseer and Kashif Zafar. "Meta-feature based few-shot Siamese learning for Urdu optical character recognition". In: *Computational Intelligence* 38.5 (2022), pp. 1707–1727. DOI: 10.1111/coin.12530.

[182] Efosa Osagie, Wei Ji, and Na Helian. "Medical Image Character Recognition Using Attention-Based Siamese Networks for Visually Similar Characters with Low Resolution". In: *Proceedings of the Third International Conference on Innovations in Computing Research (ICR'24)*. Springer Nature Switzerland, 2024, pp. 110–119. DOI: 10.1007/978-3-031-65522-7_10.

[183] Lukas Pfahler, Jonathan Schill, and Katharina Morik. "The Search for Equations – Learning to Identify Similarities Between Mathematical Expressions". In: *Machine Learning and Knowledge Discovery in Databases*. Springer International Publishing, 2020, pp. 704–718. DOI: 10.1007/978-3-030-46133-1_42.

[184] Qiqiang Lin et al. "CCLSL: Combination of Contrastive Learning and Supervised Learning for Handwritten Mathematical Expression Recognition". In: *Computer Vision – ACCV 2022*. Springer Nature Switzerland, 2023, pp. 577–592.

[185] Stephanie Lin, Jacob Hilton, and Owain Evans. *Teaching Models to Express Their Uncertainty in Words*. 2022. DOI: 10.48550/arXiv.2205.14334.

[186] Arthur Hemmer et al. *Confidence-Aware Document OCR Error Detection*. 2024. DOI: 10.48550/arXiv.2409.04117.

[187] Alexei Kaltchenko. *Assessing GPT Model Uncertainty in Mathematical OCR Tasks via Entropy Analysis*. 2024. DOI: 10.48550/arXiv.2412.01221.

[188] Peter H. A. Sneath. "The application of computers to taxonomy". In: *Journal of General Microbiology* 17.1 (1957), pp. 201–226. DOI: 10.1099/00221287-17-1-201.

[189] Andrew Howard et al. *Searching for MobileNetV3*. 2019. arXiv: 1905.02244 [cs.CV]. URL: https://arxiv.org/abs/1905.02244.

[190] Forrest N. Iandola et al. *SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size*. 2016. arXiv: 1602.07360 [cs.CV]. URL: https://arxiv.org/abs/1602.07360.

[191] Florian Schroff, Dmitry Kalenichenko, and James Philbin. " FaceNet: A unified embedding for face recognition and clustering ". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, 2015, pp. 815–823. DOI: 10.1109/CVPR.2015.7298682.

[192] Yihao Xue et al. *Investigating the Benefits of Projection Head for Representation Learning*. 2024. arXiv: 2403.11391 [cs.LG]. URL: https://arxiv.org/abs/2403.11391.

[193] David G. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". In: *International Journal of Computer Vision* 60.2 (2004), pp. 91–110. DOI: 10.1023/B:VISI.0000029664.99615.94.

[194] Marius Muja and David G. Lowe. "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration". In: *International Conference on Computer Vision Theory and Applications*. 2009, pp. 331–340. DOI: 10.5220/0001787803310340.

[195] Erik Lenas, Viktoria Löfgren, and Olof Karsvall. "Quality Prediction for Large Scale HTR: Confidence Is All You Need". In: *Document Analysis and Recognition – ICDAR*. Vol. 16972. Lecture Notes in Computer Science. Cham: Springer, 2026, pp. 450–466. DOI: 10.1007/978-3-032-36023-6_26.

[196] Van-Cuong Kieu, Florence Cloppet, and Nicole Vincent. "OCR Accuracy Prediction Method Based on Blur Estimation". In: *2016 12th IAPR Workshop on Document Analysis Systems (DAS)*. Santorini, Greece: IEEE, 2016, pp. 317–322. ISBN: 978-1-5090-1792-8. DOI: 10.1109/DAS.2016.50.

[197] Xujun Peng, Huaigu Cao, and Prem Natarajan. "Document image OCR accuracy prediction via latent Dirichlet allocation". In: *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*. 2015, pp. 771–775. DOI: 10.1109/ICDAR.2015.7333866.

Appendices

Chapter A

Appendix

A.1 PatentME-600k

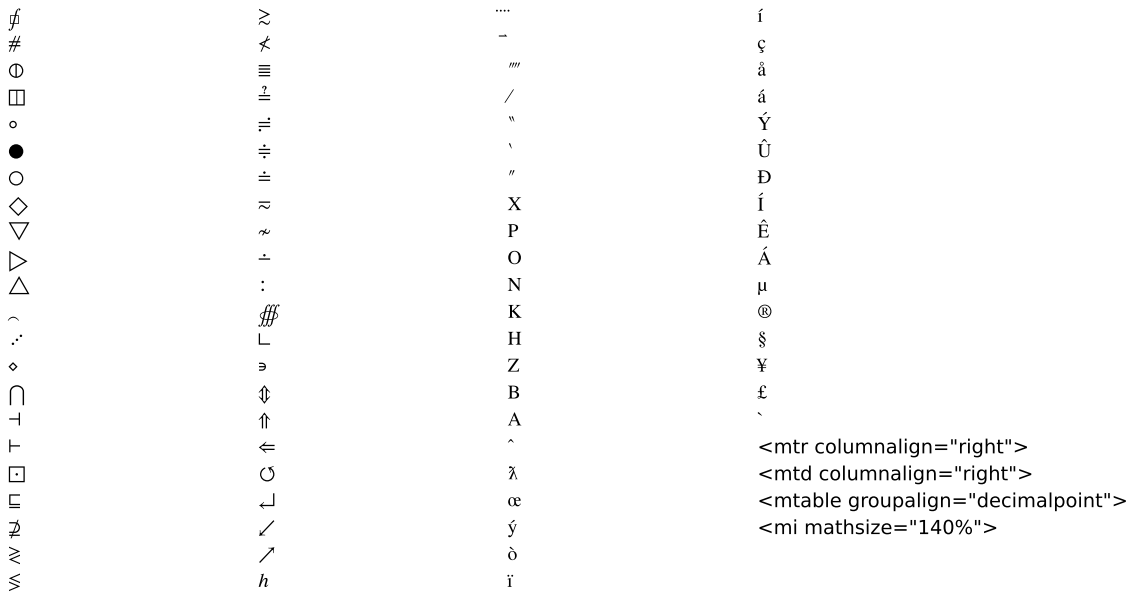


Figure A.1: List of the 86 tokens filtered from the PatentME-600k dataset. The symbols displayed as capital letters (X, P, O, N, ...) are uppercase Greek Unicode characters and not Latin uppercase characters.

Table A.1: Detailed composition of PatentME-600k.

Category	**Number of expressions**	**Percentage**
Dataset		
Total	669,582	100.00%
Split		
Train	535,665	80.00%
Validation	66,958	10.00%
Test	66,959	10.00%
Display type		
Block	442,716	66.12%
Inline	226,866	33.88%

Table A.2: Distribution of the number of mathematical expressions per patent over the 51,952 patents in PatentME-600k. The distribution is strongly right-skewed: the median patent contains four expressions, while the mean is 12.89 and the maximum is 4,874 expressions.

Statistic	**Number of expressions**
Mean	12.89
Standard deviation	39.19
Minimum	1
1st percentile	1
5th percentile	1
25th percentile	2
Median	4
75th percentile	12
95th percentile	49
99th percentile	132.49
Maximum	4,874

Table A.3: Detailed image statistics for the 669,582 mathematical expressions in PatentME-600k. Width and height are expressed in pixels. Large differences between the mean and median values indicate strongly right-skewed distributions, especially for image width, height, and area.

Statistic	**Width**	**Height**	**Aspect ratio**	**Image area**
Count	669,582	669,582	669,582	669,582
Mean	677.82	108.07	7.77	85,472
Std.	498.55	87.76	6.81	136,832
Minimum	12	24	0.04	840
1st percentile	59	47	0.63	3,481
5th percentile	83	47	1.00	5,893
25th percentile	248	59	2.55	18,894
Median	555	83	5.64	45,496
75th percentile	1,039	118	10.82	96,111
95th percentile	1,642	248	22.22	292,758
99th percentile	1,843	461	29.91	635,700
Maximum	2,811	2,846	52.97	4,801,619

Table A.4: Distribution of tokenized MathML sequence lengths in PatentME-600k. Complex MathML tags such as `<mi mathvariant="bold">` are represented as single tokens.

Statistic	**Token length**
Count	669,582
Mean	86.80
Standard deviation	96.97
Minimum	4
1st percentile	15
5th percentile	17
25th percentile	35
Median	62
75th percentile	106
95th percentile	230
99th percentile	447
Maximum	5,443

Table A.5: Tokenized MathML sequence length by dataset split.

Split	**Count**	**Mean**	**Median**	**Std.**	**Min**	**P95**	**P99**
Train	535,665	86.79	62	96.77	4	230	447
Validation	66,958	87.11	62	96.99	4	232	454
Test	66,959	86.55	62	98.55	4	231	441

Table A.6: Tokenized MathML sequence length according to display type.

Type	Count	Mean	Median	Std.	Min	P95	Max
Block	442,716	107.99	81	110.04	4	271	5,443
Inline	226,866	45.44	33	39.21	4	118	996

Table A.7: Detailed statistics of MathML structural complexity in PatentME-600k.

Statistic	# MathML nodes	MathML depth
Count	669,582	669,582
Mean	29.35	5.43
Standard deviation	34.76	2.01
Minimum	1	1
1st percentile	5	3
5th percentile	5	3
25th percentile	11	4
Median	20	5
75th percentile	35	6
95th percentile	81	9
99th percentile	160	12
Maximum	2,007	28

Table A.8: Frequency of major structural MathML elements in PatentME-600k. The percentage represents the proportion of expressions containing at least one occurrence of the corresponding element.

Element	MathML tag	Number of expressions	Percentage
Subscript	`msub`	345,042	51.53%
Subscript–superscript	`msubsup`	215,990	32.26%
Fraction	`mfrac`	173,053	25.84%
Superscript	`msup`	122,015	18.22%
Matrix	`mtable`	61,935	9.25%
Over	`mover`	54,236	8.10%
Square root	`msqrt`	26,258	3.92%
Under–over	`munderover`	19,087	2.85%
Under	`munder`	14,284	2.13%
Root	`mroot`	922	0.14%

Table A.9: Distribution of structural MathML elements per expression. The values represent the number of occurrences of each structural element within an individual expression.

Element	Mean	Std.	Min	Median	P95	P99	Max
Fraction	0.42	1.05	0	0	2	4	101
Superscript	0.38	1.37	0	0	2	5	196
Subscript	1.98	3.97	0	1	8	16	430
Subscript–superscript	0.55	1.36	0	0	2	5	158
Under	0.04	0.34	0	0	0	1	32
Over	0.17	0.86	0	0	1	4	66
Under–over	0.04	0.32	0	0	0	1	32
Matrix	0.13	0.52	0	0	1	2	58
Square root	0.06	0.36	0	0	0	1	46

Table A.10: Frequency of the 20 distinct MathML tags observed in PatentME-600k.

MathML tag	Frequency
`mi`	5,704,182
`mo`	5,359,371
`mrow`	2,448,438
`mn`	1,877,016
`msub`	1,326,302
`math`	669,582
`mtext`	436,561
`mtd`	397,882
`msubsup`	368,298
`mfrac`	281,596
`msup`	257,530
`mtr`	233,264
`mover`	114,170
`mtable`	84,160
`msqrt`	36,875
`munderover`	28,488
`munder`	23,802
`mmultiscripts`	3,618
`mprescripts`	3,196
`mroot`	1,113

A.2 Prompts Used for Vision-Language Models

The two vision-language models were evaluated using slightly different prompts. The experiments were conducted at different stages of the PhD and the prompts were not the same. The Qwen2-VL-OCR2-2B-Instruct prompt was inspired by the prompting strategy proposed in [71]. The DotsOCR prompt was kept shorter and directly specified the expected LaTeX output format.

The difference between the prompts is a limitation of the experimental protocol. However, both explicitly require the models to output only the LaTeX representation of the mathematical expression, which is the information required for the PiE-MER evaluation. The impact of prompt wording is assumed to be limited in the present comparison. A systematic comparison of multiple prompts could provide additional information, but such an experiment would require a large amount of additional inference time and computational resources for limited results.

System prompt

You are a Mathematical Expression Recognition Assistant. You will provide the LaTeX of the expression given the image. Do not transcribe the equation number. Provide ONLY the LaTeX string between $$ delimiters.

Figure A.2: Prompt used with Qwen2-VL-OCR2-2B-Instruct. The prompt was inspired by [71].

Prompt

Extract only the LaTeX content from this image. Return it between $$ ... $$ delimiters.

Figure A.3: Prompt used with DotsOCR.

A.3 Publications and Presentations Related to This Thesis

- **SIFED 2024** — Symposium International Francophone sur l'Écrit et le Document, 11 June 2024, Nantes, France. Poster presentation: *Mathematical Equations Recognition for Patent Publication*.
- **ICDAR 2024** — International Conference on Document Analysis and Recognition, 30 August–4 September 2024, Athens, Greece. Doctoral Consortium poster: *Mathematical Equations Recognition for Patent Publication and Accessibility*. Recipient of the *Best Dancer Award* (social award).
- **SIFED 2025** — Symposium International Francophone sur l'Écrit et le Document, 6 June 2025, La Rochelle, France. Poster presentation: *A Multimodal Evaluation Pipeline for Mathematical Expression Recognition: Comparisons of Datasets, Metrics, and Models*.
- **ICDAR 2025** — International Conference on Document Analysis and Recognition, 16–21 September 2025, Wuhan, Hubei, China. Published paper and poster presentation: *A Multimodal Evaluation Pipeline for Mathematical Expression Recognition: Comparisons of Datasets, Metrics, and Models*.
- **SIFED 2026** — Symposium International Francophone sur l'Écrit et le Document, 12 June 2026, Rouen, France. Oral presentation: *PatentME: A Dataset and Reference-Free Post-OCR Verification Task for Printed Mathematical Expression Recognition*. Recipient of the *Best Presentation Award*.
- **ICDAR 2026** — International Conference on Document Analysis and Recognition, 30 August–4 September 2026, Vienna, Austria. Published paper and oral presentation: *PatentME: A Dataset and Reference-Free Post-OCR Verification Task for Printed Mathematical Expression Recognition*.

FOLIO ADMINISTRATIF

THESE DE L'INSA LYON, MEMBRE DE L'UNIVERSITE DE LYON

NOM : **WIECKOWIAK** DATE de SOUTENANCE : **15/12/2026**

Prénoms : **François**

TITRE : **Amélioration de la précision de la reconnaissance des expressions mathématiques imprimées pour la publication de brevets**

NATURE : **Doctorat** Numéro d'ordre :

(donné au moment de l'autorisation de soutenance)

École Doctorale : **Informatique Mathématiques**

Spécialité : **Informatique**

RÉSUMÉ :

Pour permettre la publication des brevets auprès de l'Office européen des brevets, l'entreprise Luminess est chargée de normaliser des demandes de brevet reçues dans des formats très hétérogènes en documents XML et PDF standardisés. Ces documents sont ensuite rendus accessibles à l'ensemble de la communauté scientifique et peuvent être facilement recherchés et retrouvés grâce à une annotation précise et normalisée de leur contenu. Cependant, des erreurs de reconnaissance peuvent altérer ces annotations et empêcher la recherche ou la récupération de documents pertinents.

Cette thèse porte sur l'automatisation de la reconnaissance des expressions mathématiques dans les demandes de brevet. Ces expressions, particulièrement complexes à reconnaître automatiquement, sont actuellement transcrites et vérifiées manuellement à plusieurs reprises afin de satisfaire aux exigences de qualité de l'Office européen des brevets. L'objectif de cette thèse est ainsi de réduire la charge associée à leur transcription et à leur vérification, tout en maintenant un niveau de qualité supérieur ou égal à 99,99%.

Une première contribution consiste en un protocole d'évaluation multimodale permettant d'analyser les performances des systèmes de reconnaissance optique de caractères selon différentes représentations des expressions mathématiques. Cette évaluation met en évidence les limites des approches existantes sur les documents de brevet et conduit à la création de PatentME-600k, un jeu de données de plus de 600 000 expressions mathématiques extraites de brevets. Sur cette base, MML-Net, une architecture Vision Transformer encodeur-décodeur, est proposée pour la reconnaissance directe des expressions mathématiques en MathML. Le modèle atteint un score de similarité de Levenshtein de 93,7% et dépasse les performances de l'état de l'art sur cette tâche.

Afin de permettre l'intégration de MML-Net dans une chaîne de production industrielle, plusieurs briques complémentaires sont développées. Un ordonnanceur basé sur un Random Forest et des caractéristiques visuelles sélectionne les expressions pour lesquelles le modèle est susceptible de produire une transcription correcte, afin de réserver l'intervention humaine aux cas les plus complexes. Un vérificateur post-OCR, basé sur une architecture siamoise et entraîné spécifiquement sur les erreurs de MML-Net, détecte ensuite automatiquement les prédictions susceptibles d'être incorrectes et les oriente vers une validation humaine. Enfin, une contribution porte sur l'assistance à la saisie et la réduction des erreurs introduites lors des opérations manuelles. Différentes stratégies d'intégration de ces briques dans la chaîne de production de Luminess sont étudiées afin de combiner reconnaissance automatique et validation humaine, d'augmenter le niveau d'automatisation et de préserver les exigences de qualité du processus industriel.

MOTS-CLÉS : **Reconnaissance d'Expressions Mathématiques, Reconnaissance Optique de Caractères, Évaluation Multimodale, Vérification Post-OCR, Apprentissage de Métriques, Demandes de Brevet.**

Laboratoire(s) de recherche : **LIRIS**

Directrice de thèse : **Véronique EGLIN**

Président du Jury : **(président, nommé au moment de la soutenance)**

Composition du Jury :

Harold MOUCHÈRE, Mickaël COUSTATY, Bertrand COÜASNON, Véronique EGLIN, Tony BONNET, Laëtitia ROUSSEAU, Stéphane BRES