A Survey on Handwritten Mathematical Expression Recognition: The Rise of Encoder-Decoder and GNN Models

Thanh-Nghia Truong[1], Cuong Tuan Nguyen[2], Richard Zanibbi[3], Harold Mouchère[4], and Masaki Nakagawa[1]

[1]Tokyo University of Agriculture and Technology, Tokyo, Japan
[2] Vietnamese-German University, Binh Duong, Vietnam
[3]Document and Pattern Recognition Lab,
Rochester Institute of Technology, New York, USA
[4]Nantes Université, École Centrale Nantes, CNRS, LS2N, UMR 6004, F-44000 Nantes, France
fw7852@go.tuat.ac.jp, cuong.nt2@vgu.edu.vn, rlaz@cs.rit.edu,
harold.mouchere@ls2n.fr, nakagawa@cc.tuat.ac.jp

Abstract

Recognition of handwritten mathematical expressions (HMEs) has attracted growing interest due to steady progress in handwriting recognition techniques and the rapid emergence of pen- and touch-based devices. Math formula recognition may be understood as a generalization of text recognition: formulas represent mathematical statements using a two dimensional arrangement of symbols on writing lines that are organized hierarchically. This survey provides an overview of techniques published in the last decade, including those taking input from handwritten strokes (i.e., 'online', as captured by a pen/touch device), raster images (i.e., 'offline,' from pixels), or both. Traditionally, HMEs were recognized by performing four structural pattern recognition tasks in separate steps: (1) symbol segmentation, (2) symbol classification, (3) spatial relationship classification, and (4) structural analysis, which identifies the arrangement of symbols on writing lines (e.g., in a Symbol Layout Tree (SLT) or LaTeX string). Recently, encoder–decoder neural network models and Graph Neural Network (GNN) approaches have greatly increased HME recognition accuracy. These newer approaches perform all recognition tasks simultaneously, and utilize contextual features across tasks (e.g., using neural self-attention models). We also discuss evaluation techniques and benchmarks, and explore some implicit dependencies among the four key recognition tasks. Finally, we identify limitations of current systems, and present suggestions for future work, such as using two-dimensional language models rather than the one-dimensional models commonly used in encoder–decoder models.

Keywords: Mathematical Expression Recognition; Handwriting Recognition; Symbol Recognition; Spatial Relationship Classification; Structural Analysis; Deep Neural Networks; CROHME competitions; public dataset

1 INTRODUCTION

Mathematical expressions or mathematical formulas have an essential role in scientific documents, and are an indispensable tool for describing problems, theories, and solutions in mathematics, physics, and many other fields. Writing math formulas by hand is natural and convenient, and with the rapid emergence of pen- and touch-based input devices such as digital pens, tablets, and smartphones, handwriting is often used for math input. However, high recognition accuracy is required for users to utilize handwritten mathematical expression (HME) input on computers.

HME recognition has been studied for many decades, starting with early proposals to apply top-down parsing of attributed two-dimensional mathematical expression grammars by Anderson [1], bottom-up parsing of expression grammars using the *dominance* of operators over their arguments by Chang [2], and later syntactic method by Belaid and Haton [3]. In the last decade,

research on HME recognition has increased greatly: this is due to a combination of factors, including the widespread use of pen/touch-based input devices, the ease of inputting mathematical formulas using handwriting on these devices, the Competition on Recognition of Online Handwritten Mathematical Expressions (CROHME) [4, 5] that has provided a standard evaluation benchmark with public datasets and evaluation tools, and progress in deep learning techniques.

In this paper, we provide a survey of HME recognition techniques developed over the last decade, during which HME recognition accuracy has greatly increased. We focus in particular on the emergence of deep neural network methods, and particularly the encoder–decoder models that currently obtain state-of-the-art accuracy, along with emerging Graph Neural Network (GNN) approaches. We also make reference to older research [6, 7, 8] because significant progress was made in earlier methods using structural approaches (i.e., grammar-based, tree-based, and graph-based) on publicly available datasets. Recently, Zhelezniakov et al. [9] conducted a survey of online HME recognition considering systems, user interfaces, and applications from a broader perspective. In contrast, this survey provides an in-depth survey on recent HME recognition methods, and we believe that the two surveys are complementary.

Before the last decade, most HME recognition methods were evaluated using private datasets that are not publicly available. However, over the last decade, many datasets have been collected and are publicly available. The CROHME series of competitions has provided an open platform and benchmark for HME recognition since 2011. As a result, later in the paper recognition rates for earlier methods are reported using private datasets, whereas those of recent methods are reported using public datasets, so that a fair comparison between modern techniques can be made.

Input Modalities: Online vs. Offline. Generally, there are two input modalities for handwriting. One uses sequences of pen-tip or finger-top coordinates collected from modern electronic devices, referred to as *online* patterns. Online expressions are created using a pen-tip or finger-top represented by *(x, y, timestamp)* coordinate triples. A sequence of coordinates from a pen/finger press ('pen/touch-down') to the pen/finger lift ('pen/touch-up') is called a *stroke*. The second modality is images of handwriting captured by a scanner or camera; these are termed *offline* patterns.

Handwritten strokes are represented explicitly in online data by pen/touch-up and pen/touch-down event symbols. However, segmenting symbols in online strokes, even with associated timestamps, is complicated by variations of stroke orders, including delayed strokes (e.g., adding the dot to an 'i' after completing another symbol) and strokes that users add to replace missing and extend partially sampled strokes. On the other hand, while offline images are free from stroke order variations and stroke duplications, the segmentation of strokes into symbols is still required unless a 'segmentation-free' method that considers the entire image when identifying symbols is used (i.e., a *holistic* method). Offline images are easily created from online strokes by drawing a polyline between successive coordinates from pen/touch-down to pen/touch-up. This conversion of strokes to images is commonly performed in HME recognition [10, 11, 12] to avoid the stroke ordering and duplication issues encountered when recognizing using sequential online stroke data, and to generate visual features for recognition.

Sequential recognition models such as the Hidden Markov Models (HMMs), time-delayed Neural Networks [13], and Recurrent Neural Networks (RNNs) have been applied to recognizing online input directly, whereas spatial signal processing methods such as discriminant functions, Support Vector Machine (SVM), and Convolutional Neural Networks (CNNs) have been used for recognition in offline HME images [10]. Both stroke-based sequential models and image-based spatial recognition models can be used for online handwritten input because of the ease of converting handwritten strokes into images. In this survey, although we present and discuss methods covering both online and offline inputs, we emphasize recognition models using online

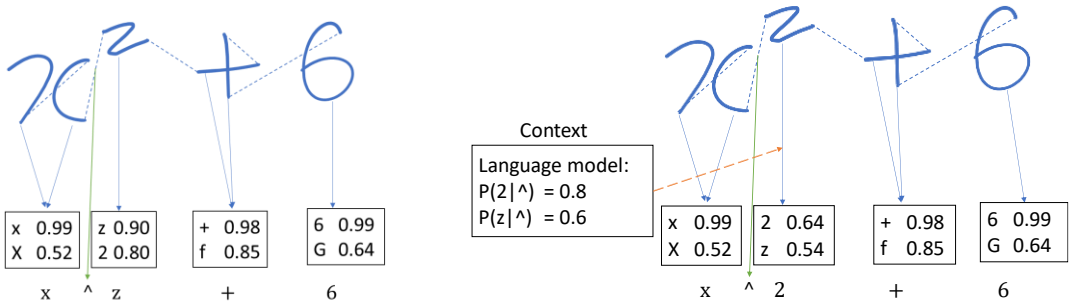
strokes for the following reasons. First, a smaller number of methods have been reported for offline input, and extensive studies have been conducted on online methods in the last decade to provide natural user interfaces for pen/touch-sensitive devices. Moreover, methods originally developed for online inputs have recently been adapted for offline inputs.

A math formula is a complex two-dimensional arrangement of various symbols, numerals, and operators, along with the spatial relations between them. HME recognition involves four structural pattern recognition problems: (1) symbol segmentation, (2) symbol classification, (3) spatial relationship classification, and (4) structural analysis (possibly including the use of an expression grammar). All four tasks have inherent ambiguities: For example, several symbols have similar or identical shapes in handwriting (e.g., {1, l, | } and {o, 0, O}), there are many style variations and distortions in handwriting, and there is often sampling noise in both handwritten strokes and stroke/formula images. To reduce uncertainty, contextual analysis and measures of the total likelihood of a recognized formula are used in HME recognition. Contextual features may take many forms, such as geometric, statistical, syntactic, and linguistic (i.e., language model-based).

Utilizing Context in HME Recognition: An Example. Figure 1 shows an example of *online* (stroke) input for the formula "$x^2 + 6$", drawn using six handwritten strokes: Solid lines visualize the drawn strokes, while dashed lines visualize the sequence of pen/touch-up and pen/touch-down events after the leftmost stroke of the 'x' is drawn.

Figure 2 illustrates recognition with and without contextual analysis. In this example, we have already correctly segmented the strokes into symbols (i.e., the two strokes of the 'x' and '+' are merged), and correctly identified spatial relationships between symbols (i.e., "$x + 6$" lie on the main writing line, and there is a superscript "x^2"). In Figure 2, each symbol has two candidate class labels with scores (Note: real systems will consider additional labels). Without contextual analysis, we select all symbol labels with the highest score independently, producing "$x^z + 6$". However, if we add context in the form of a simple language model capturing the probability of a symbol based on its spatial relationship, we find that '2' rather than 'z' is more likely as a superscript, and we then correctly recognize the HME as "$x^2 + 6$". We will revisit this example again later in the paper.

Fig.1. HME pattern for "$\boldsymbol{x^2 + 6}$." Stroke ordering is shown by dashed lines from each pen/touch-up (*lift*) event to the next pen/touch-down (*press*) event.



(a) Recognition without contextual analysis ($x^z + 6$). (b) Recognition with contextual analysis ($x^2 + 6$).

Fig. 2. Recognitions without context analysis and with context analysis. The top two class candidates with scores are shown below each symbol.

Organization of this Paper. The remainder of this paper is organized as follows. Section 2 surveys the structural approach used in the first half of the past decade. Section 3 surveys the DNN approach, specifically encoder–decoder models and Graph Neural Networks (GNNs)-based models. Section 4 presents further advances in the DNN approach. Section 5 summarizes data augmentation and generation. Section 6 presents an evaluation of the HME recognition methods using available public datasets. Section 7 discusses the achievements over the past decade and the remaining issues. Section 8 discusses related research topics, and finally Section 9 concludes the paper.

2 Structural Recognition Approaches

In the early part of this past decade, grammar-based [14], tree-based [15], and graph-based methods [16] were commonly used for HME recognition. We refer to these methods as *structural* because they construct a graph representing symbols and/or subexpressions as nodes with explicit input regions associated with them (e.g., symbols are defined as groups of strokes (online) or specific image sub-regions (offline)). Edges in the graph represent relationships between symbols and/or sub-expressions. These relationships may be spatial such as for the superscript illustrated in Figure 2, or to identify mathematical entities and their relationships, such as for the type and/or position of arguments in operations within an HME.

Grammar-based methods make use of a math expression syntax, e.g., in the form of a context-free grammar, to constrain the space of possible formula interpretations. *Tree* and *Graph* based methods do not use an explicit grammar or language definition, but still construct a formula representation from specific input elements/regions using a fixed vocabulary of symbols and relationships. While tree and graph-based methods use fewer constraints than grammar-based methods, they do constrain their outputs to avoid invalid HME interpretations (e.g., constraining output graphs to be rooted trees).

A limitation of both traditional grammar-based and tree/graph-based structural methods is that recognition subtasks are often executed independently, with their results combined to produce complete formula interpretations in later processing. This limits accuracy by underutilizing context, and by optimizing recognition for subtasks independently. Despite this, some commercial systems have used structural approaches in real systems and applications, such as in systems created by MyScript [17], Wiris [18], and Samsung [19].

In general, four subtasks are involved in both online and offline HME structural recognition [6, 5, 20]: symbol segmentation, symbol classification, relation classification, and structural analysis. We summarize the four structural tasks in HME recognition, and how they are commonly approached in structural recognition approaches below.

1. **Symbol segmentation** identifies the location of symbols in the input (e.g., stroke sets or image regions). Commonly, these models utilize binary segmentation scores to merge/split strokes or image regions (e.g., connected components) into symbols. This binary segmentation of strokes/sub-images is often performed using geometric features, but it is difficult to segment symbols accurately without also considering their class labels and spatial relationships. As a result, many structural approaches generate multiple segmentation hypotheses and then select the best segmentation while considering symbol classification and/or structural analysis results. A detailed description is provided in [9].
2. **Symbol classification** identifies the label/type of a symbol represented by a set of strokes (online) or image region (offline). Normally, candidate labels with recognition scores for segmented symbols are generated and then selected from when producing the final interpretation for an HME. Recognition methods can be broadly classified into four categories: structural, statistical, DNN-based, and hybrid methods. Symbol recognition techniques used in structural HME recognizers has been well-documented in previous surveys [6, 7, 8].

3. **Relationship classification** determines the type of spatial relation between a pair of symbols or subexpressions (represented by a parse tree or formula graph sub-graph). Spatial relations are often classified into six classes: horizontal adjacency, subscript, superscript, over, under, and inside (e.g., for square roots). For spatial relation classification, the bounding box information of symbols is typically used to identify spatial relationships [14, 21, 22]. Alvaro introduced the revised centroid for representing symbol locations in relationship detection [14], while Le et al. introduced a revised bounding box called a body box [21] for symbol locations, which is trainable but assumes correct symbol classification. Some studies have utilized the shapes of symbols for relation classification using histograms of directions [23] or shapes [24].
4. S**tructural analysis** utilizes the three tasks above as part of a larger process to interpret the spatial arrangements of candidate symbols in an HME and produce candidate formula structure interpretations. This is often done using grammatical and/or contextual constraints to insure syntactically valid mathematical formulas in the output as described earlier in this Section. Structural analysis has been extensively studied over the past decade. Very often, geometric and linguistic contexts are explicitly incorporated into functions for pruning and scoring formula interpretations [23, 19].

In many structural approaches, tasks proceed sequentially, or earlier tasks proceed in parallel, with later tasks providing feedback to the earlier tasks [25]. For example, symbol segmentation can be performed once and fixed or combined with symbol classification. As another example, spatial relation classification is often integrated within structural analysis.

To reduce variance in input data, many systems perform additional tasks such as preprocessing, normalization, and noise reduction. For example, most online methods use stroke resampling and smoothing [11, 20], whereas offline methods often use connected component analysis [26] and geometric feature extraction [27] to preprocess raw input patterns. Normalization of data is often applied to make later processing easier, including center normalization [28], writing speed normalization, and size normalization [14] for online strokes, and intensity normalization and moment normalization for offline HME images [29, 30]. Noise reduction is also commonly applied to improve input quality.

However, even with preprocessing and normalization, for both online and offline recognition, the input may contain many small symbols (e.g., dots, commas, and diacritical marks) that are difficult to distinguish from noise [6]. Preparing and/or learning Language Models (LMs) can help address these issues by representing probabilistic contextual constraints for structural analysis, as seen previously in Figure 2. LMs were used in commercial systems created by Myscript and Samsung [20, 19].

2.1 Grammar-Based (Syntactic) Methods

In formal language theory, a variety of string-based languages may be defined using *grammars*. Expression grammars for mathematical expressions, which most commonly capture the hierarchy of operations represented in a formula expression (e.g., in a programming language) are perhaps one of the most common examples of formal languages. In particular, *context-free grammars* are widely used to define math expression grammars. In HME recognition models, expression grammars are used to constrain candidate symbol segmentation hypotheses, symbol classes, and spatial relations so that output formulas represent *valid* formulas, according to a chosen expression grammar.

When a chosen grammar matches expected input well, it can help improve recognition performance by pruning the space of possible outputs and reducing uncertainty. One challenge is when inputs do not match the grammar: in that case, syntactic constraints may cause mis-interpretation, or even parse failure if output formulas are required to match the expression grammar used by a recognition system.

In the very early days of HME recognition, Anderson proposed a method where every grammar rule is associated with partitioning predicates that evaluate partitions of input symbols [1]. The parser created an operator tree from top to bottom according to an expression grammar, which was later converted into a string using the bottom-up synthesis step. Much more recently, MacLean et al. introduced a grammar-based approach that used relational grammar and fuzzy sets [31]. Their parser, based on fuzzy set logic, constructed a shared-parse forest and the interpretation was extracted from the parse trees.

A standard *context-free grammar (CFG)* contains a set of derivation rules with no weights or prioritization. Thus, all possible parse candidates are considered equally likely. Yamamoto et al. utilized *stochastic* CFGs (SCFGs) to estimate probabilities for each rule of the grammar [22]. Candidate parse trees are scored using the product of the (independent) rule probabilities, allowing candidates to be ranked by likelihood. Prusa and Hlavac proposed a two-dimensional CFG (2D-CFG), which is defined by adding a finite set of relations between two elements to represent two-dimensional languages such as mathematical expressions [26]. Awal et al. used 2D-CFG with global optimization for symbol recognition and relation classification scores [32]. Much earlier in the 1980s, Chou added rough probability estimates to a 2D-SCFG [33], which is where this approach was first attempted.

Table 1: A Simple Grammar for Math Formula Symbol Layouts

Grammar rules
<Let> →{ a | b | c | … | x | y | z }
<Num> → { 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 }
<Ope> → { + | - | × | / }
<SupExp> → { <Let> superscript <Num> | <Let> superscript <Let> }
<Term> → { <Let> | <Num> | <SupExp> }
<LeftExp> → <Term> horizontal <Ope>
<Exp> → <LeftExp> horizontal <Term> | <Term>

Let us consider a simple example of a CFG for representing symbol layout in HMEs, as shown in Figure 1, which represents a rather limited language of math expressions. Letters represented by the *non-terminal* (variable) <Let> must be lower-case letters. Numerals (<Num>) are "0" to "9'. Operator symbols (<Ope>) are among "+", "-", "×" and "/". A superscript expression ＜SupExp＞ is a letter with either a number or letter as its superscript. A term <Term> may be a letter, numeral, or superscript expression (<Let>, <Num>, or <SupExp>). An expression <Exp> is composed of a <LeftExp> (term followed by an adjacent operator) or a single <Term>.

We can extend our CFG to a *Stochastic Context-Free Grammar* (SCFG) by associating each grammar rule with a probability of applying that rule. For example, <Let> may produce "a", "b", "c", … , "x", "y" or "z" with the same probability if they occur equally often or with probabilities values if some letters more frequently seen than others when training the LM. Normally, in an SCFG, all rules with the same non-terminal symbol on the left-hand-side have probabilities that sum to one, representing a probability distribution of possible replacements for each non-terminal symbol.

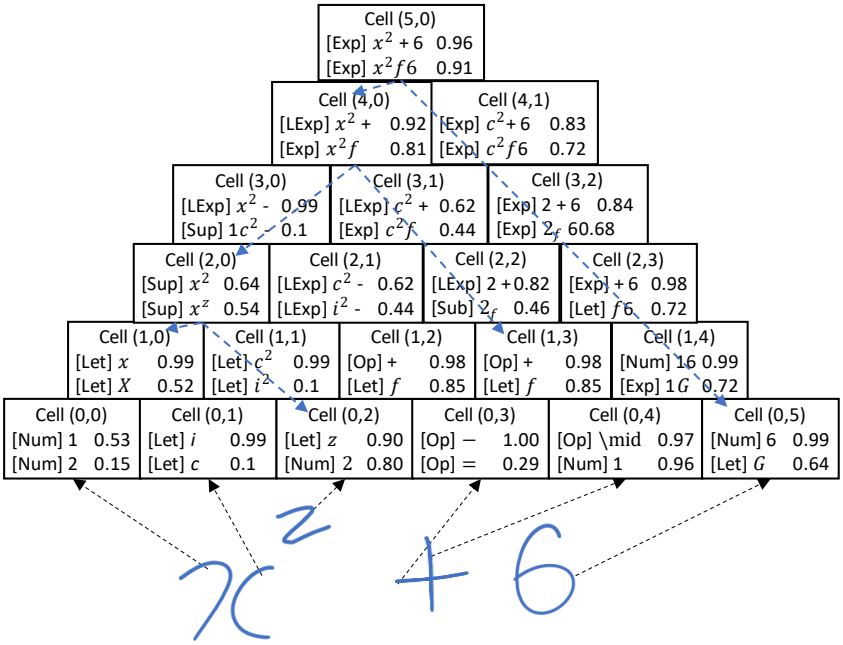


Fig. 3. Parsing table for HME denoting "$\boldsymbol{x^2+6}$."
Each cell shows its index as *(length - 1, starting-stroke-index)*, candidate non-terminal symbols, and top two parse candidates.
Dashed blue lines illustrate the interconnection of cells through grammar rules for the highest probability parse.

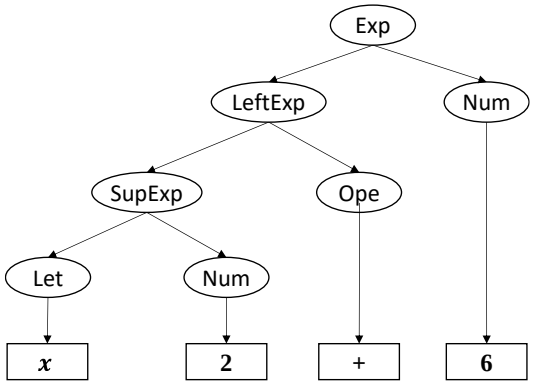


Fig. 4. Parse tree for "x^2+6" extracted as the highest-probability interpretation from the parse table in Fig 3.

Figure 3 illustrates parsing the online HME in Figure 1 ("x^2+6") using a SCFG. We construct a parsing table with the selected parse tree with the highest probability shown in Figure 4. Each handwritten stroke is assigned a list index position in the input. Strokes may be a symbol on their own (e.g., for "6") or part of a symbol (e.g., "+" is drawn using two strokes). The parse table is organized in a pyramid, with stroke input sequences of increasing length from the bottom row (1 stroke) to the top row (all 6 strokes). Each cell holds partial parse trees obtained using grammar rules from Table 1.

At each cell, the two most-likely results as determined by rule probabilities are shown; this includes classification probabilities for individual symbols for *terminal* rules (see bottom two rows), and a probability combining symbol segmentation, symbol classification, relation classification, and grammar rule probabilities in the other rows. Each cell is identified by pair (*i*, *j*) representing the number of strokes (*i*) past the first input stroke (*j*) used to produce parse trees in the cell. Dashed arrows represent

the highest probability parse tree ("$x^2 + 6$") in the topmost cell (5,0), where the parse tree root nonterminal for an expression containing all input strokes (<Exp>) is produced. Dashed blue arrows in the table correspond to the parse tree in Figure 4.

The table is filled bottom-up. First, each of the six strokes ($0 <= j <= 5$) is assigned candidate class labels as shown in the bottom row. Labels and the non-terminals (e.g., <Num>, <Let>) that generated them are shown. Each next level is constructed by applying rules for any two cells below that partition the input sequence represented in the cell pair (i,j). At level *i=1*, this will be either a terminal rule (symbol) for two adjacent strokes *or* the result of applying a production rule (e.g., at Cell (1,4) the expression '1G' is a candidate). At level *i=2*, the best recognition candidate in Cell(2, 0) is "x^2", which combines Cell(1, 0) "x"/ "*X*" and Cell(0,2) "2" / "*z*". In this case, the combination of "x" and "2" with the superscript relationship has a higher estimated probability than "x" and "z" using rules associated with the non-terminal <Sup>.

This parsing algorithm is a variation of the well-known Cocke-Younger-Kasami (CYK) dynamic programming algorithm used to generate all legal parses of an input sequence for a given context-free grammar. We are using a stochastic version that also computes probabilities and ranks possible derivations. The complexity of CYK is $O(n^3|G|)$, where *n* is the length of the input sequence, and |G| is the number of rules in grammar G.

For HME recognition, SCFGs have been used to parse from strokes as shown in Figure 3 [22, 34], or symbols [14, 35]. At the stroke level, rules are designed to model the likelihood of handwritten strokes in time order, in some cases with added production designed to handle stroke/symbol order variations. In one approach [21], production rules were provided for handling different input orders for the numerator, denominator, and horizontal line in a fraction. Parsing from the symbol level, CYK has been used to determine the most likely symbol labels and spatial relationships as constrained by an SCFG.

Some other approaches have used more structurally complex *graph grammars* to resolve ambiguities in an HME graph or a subgraph [36, 37]. Graph grammars contain graphs on their left and right-hand sides in their rules. Previously, Kosmala et al. used context-dependent graph grammar [36], whereas Aguilar et al. used a context-free graph grammar [37] to construct parsing trees for preliminary HME graphs.

Grammar-based methods have some additional drawbacks. First, for methods using CYK parsing of SCFGs, the run-time complexity of $O(n^3|G|)$ can result in slow execution, particularly for large inputs and large grammars. As a result, often strong constraints are placed upon spatial ordering and distances to reduce computation [14]. These constraints limit possible segmentations and relationships, for example, by removing pairs of objects with a blocked Line-of-Sight between them. Second, grammar rules must be carefully defined to capture the types of HMEs to be recognized, and this generally needs to be done manually by trial-and-error, and there is no 'math grammar' to capture all domains of mathematics. Third, it is difficult to design a function that can optimize the recognition score: Even if the correct interpretation is among candidates, it is difficult to reliably combine estimated probabilities/scores from symbol segmentation, symbol classification, relation classification, and grammar production rules, each of which is usually produced by a different classifier with a different scale.

2.2 Graph-Based Methods

In graph-based methods, the symbols and structure of a math formula are recognized, but without the use of an explicit expression grammar. The only 'grammar' is in the form of the sets of class labels for symbols and relationships represented using the nodes and edges of a graph, respectively, along with constraints on the output graph (e.g., that the graph must be a rooted

tree). Normally, multiple interpretations are evaluated by combining segmentation, symbol classification, and relation probabilities, and the interpretation with the highest probability is selected as the recognition result.

Because math formulas have hierarchical structure in both their appearance and mathematical syntax (i.e., how operations are applied to arguments), graph-based methods generally produce a Symbol Relation Tree (SRT) representing the placement of symbols on writing lines and their spatial arrangement (e.g., as represented in LaTeX), or an Operator Tree (OPT) representing the mathematical operation syntax with arguments at the leaves, and operations in the internal nodes of the tree. Note that the parse trees produced by grammar-based methods are roughly equivalent to SRTs or OPTs (with additional non-terminal nodes), depending upon whether the grammar represents appearance (e.g., see Table 1) or mathematical expression syntax.

Many early graph-based methods for online HME recognition [38, 39, 40, 41] were based on probability maximization or penalty minimization. These methods share a common approach: all possible symbol hypotheses are computed from symbol segmentation and classification scores. An advantage of these methods is that local recognition errors can be corrected when evaluating the full recognition tree (i.e., through contextual information). A first example of this was described by Eto et al., in which they proposed a graph penalty minimization method for recognizing offline *printed* formulas [28]. The total penalty is the sum of the penalties on edges and a global penalty term used to enforce contextual constraints. In this work, the recognition problem is transformed into determining the minimum spanning tree (MST) in a graph of hypothesized symbols and relationships. Later on, Hu et al. used an MST-based parser applied to a directed Line-of-Sight (LOS) graph over strokes, in which an edge is present only if there is an unobstructed path from the center of a stroke to the convex hull of another stroke [42]. After classifying LOS edges (merge/split) to segment symbols and then classifying detected symbols, a second LOS graph is constructed over the detected symbols, and the edges of this symbol-level graph are classified using spatial relationships. Finally, a *directed* MST capturing symbol layout as an SRT is constructed using Edmonds' spanning arborescence algorithm. Shah et al. improved the previous work by enhancing the visual features representations using additional visual context from the LOS graph [43].

Zhang et al. proposed a tree-based Long Short-Term Memory (LSTM) neural network to jointly learn symbol and relation classification [15]. They used a Stroke-Label graph (SLG) to represent an online HME in which nodes represent strokes, whereas labels on the edges encode either segmentation or layout information. Similarly, Truong et al. used an LSTM-based temporal classifier to build a graph-based online HME recognition system [44].

A clear advantage of graph-based methods is that the graph can be used without major revisions to expand the types of HMEs that can be recognized, simply by expanding the set of symbol and relationship classes. However, they share the disadvantage of the complexity in defining scoring mechanisms to combine sub-tasks, and some such as the model by Hu et al. [42] lack the contextual constraints that expression grammars and statistical language models can use to increase accuracy.

2.3 Other Approaches and Incremental Recognition

In addition to grammar-based and graph-based methods, many researchers have attempted other original ideas to discover new ways to recognize HMEs. For example, Zanibbi et al. used a set of tree transformations and grammar rules to create an SRT followed by an operator tree using a compiler-style architecture [27]. An operator tree can be used to evaluate an expression, or to translate an expression into the format of a computer algebra system (e.g., Maple or Mathematica). MacLean et al. proposed a probabilistic tree-scoring function, in which a Bayesian network was constructed to capture and organize all recognizable

interpretations [45]. Rhee et al. proposed a layered search tree, in which the costs of symbol recognition and spatial relation classification were estimated from a set of predefined heuristic predictions [46].

Most of the aforementioned methods are batch recognition methods that recognize an online HME after it has been entered completely. This is the simplest way to obtain a high recognition rate because all contextual information for the formula is available. However, in user-facing applications, if recognition is performed after the user has finished writing this can incur a long waiting time, particularly when a formula is large. To reduce waiting time, many researchers have proposed incremental recognition methods for online HMEs. These incremental methods provide recognition results after each new stroke or a new sequence of strokes. A key problem is how to infer context for accurate recognition when a formula is incomplete and perhaps not a legal mathematical expression. With incremental recognition, user waiting times decrease, but recognition accuracy may be sacrificed owing to limited contextual information [47].

For incremental recognition, *stroke-order-dependent* methods incrementally construct interpretations. MacLean et al. updated a shared-parse forest whenever a new stroke was entered [31], while Phan et al. incrementally built a parsing table after every new stroke [47]. This was further improved by revising previous segmentations and recognitions after more of a formula had been entered [48]. In contrast, *stroke-order-free* methods focus on updating only subexpressions altered by new strokes. Predovic et al. placed a new input stroke into a stroke region [49] and updated corresponding parsing-table cells affected by the new stroke [50]. Vuong et al. assigned the latest stroke to the corresponding position in a tree-based HME interpretation (SRT), and performed a progressive structural analysis for dynamic recognition [51].

3 Deep Neural Network (DNN) Approaches

Structural recognition approaches often deal with the four subtasks of HME recognition separately. As seen earlier in Figure 2, such approaches often lead to ambiguities that can be resolved through context (e.g., across recognition tasks). Moreover, grammar-based (syntactic) methods such as SCFGs require the manual definition and empirical validation of expression grammar rules for new structures, and both the explicit language models in a grammar and implicit language models defined by node and edge labels and constraints in graph-based methods may over- or under-constrain the space of possible formula interpretations.

To avoid designing expression grammars while still discovering contextual constraints that improve recognition accuracy, it would be preferable to use statistical learning methods that address all four HME recognition subtasks *simultaneously*, and share information between them. For this purpose, researchers have used deep neural networks including Convolutional Neural Networks (CNN), Long-Short Term Memory networks (LSTM), their bi-directional variants (BLSTM), and Transformer networks to solve a variety of computer vision and related tasks. CNNs, LSTMs/BLSTMs, and Transformers have demonstrated their strengths in solving automatic segmentation, detection, and recognition problems [52, 53].

Over the last decade, state-of-the-art structural HME recognition models gave way to new DNN-based models that used more integrated contextual information across tasks. These approaches are generally trained from strokes (online) or images (offline), with only an initial set of symbol and relationship classes for SRTs, along with target expression representations for use in training in either a string representation (e.g., LaTeX) or graph representation. No expression grammar is used, and training samples are *(i,o)* pairs of inputs and their associated target SRT (string/graph) outputs. Learned network weights across layers and in context vectors capture contextual patterns in correlations between input features and target outputs. Where sufficient

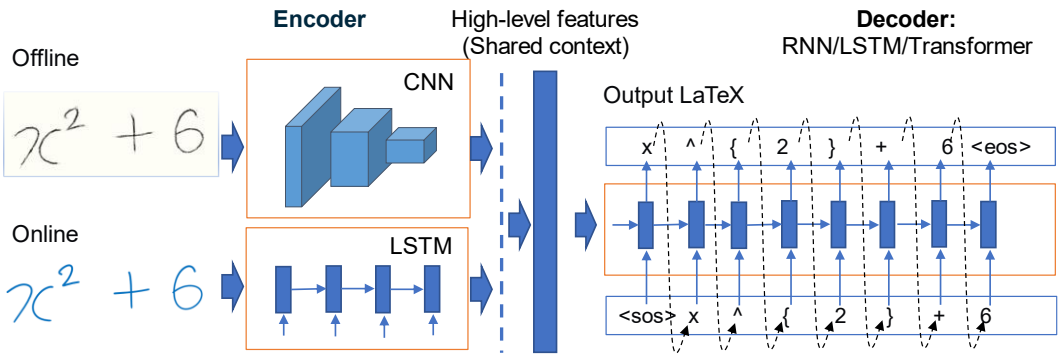


Fig. 5. General architecture of an encoder–decoder model. Dashed arrows show output of previous steps being used as input for the next step. <sos> - start of string, <eos> - end of string.

training data is available, DNN-based approaches have produced large gains in accuracy over previous structural models, mirroring progress in other pattern recognition tasks.

In the remainder of this section, we focus on the encoder–decoder models that currently obtain state-of-the-art performance, as well as more recent Graph Neural Network (GNN)-based techniques, which are competitive with encoder–decoder models, While GNN techniques require explicit graphs for use in training, they have the benefit of more interpretable models.

3.1 Encoder–Decoder Models

The encoder–decoder framework has been successfully applied to several fields, such as parsing [54], speech recognition [55], machine translation [56], and HME recognition [10, 11, 57]. This approach jointly handles all four recognition tasks identified in Section 2: (1) symbol segmentation, (2) symbol classification, (3) classification of spatial relations, and (4) structural analysis during recognition. It does so by treating the recognition task as a *sequence-to-sequence* problem, e.g., from a time-ordered list of strokes in the input being mapped to a LaTeX string representing an SRT. A number of encoder–decoder models [10, 11, 12, 57] have obtained strong results including state-of-the-art accuracy for HME recognition. An analysis of their results on benchmarks is presented in Section 7.

As seen in Figure 5, encoder–decoder models for HME recognition have two main steps: one generates an embedded feature representation of the input, and a second recurrently generates a sequence of output characters for a string describing an SRT (e.g., in LaTeX). In Figure 5, the tokens <eos> and <sos> represent 'end of string' and 'start of string,' respectively.

These models are composed of jointly optimized components that provide optimization using global as well as local information. Joint optimization allows the model to learn to extract high-level features, which provide shared context for all four HME recognition subtasks. Here, shared context refers to geometric and linguistic context information that is implicitly learned and propagated during training, and then utilized during recognition/inference for new inputs. Some systems also use additional context such as language models to improve their performance [11, 58]. Encoders may be an LSTM/BLSTM network (for online data) or a CNN (for offline data) that accepts an input pattern and encodes high-level features (e.g., a shared context vector) from the input data. The decoder, which is typically a recurrent neural network (e.g., LSTM, or more recently a Transformer) generates an output character sequence describing the input HME.

The encoder model generates high-level features passed to the decoder in a sequence. The decoder learns a mapping between feature sequences and LaTeX character sequences through a neural attention mechanism. As shown in Figure 5, the decoder also incorporates the context of the previously decoded inputs (i.e., output characters) to predict the next output character. During

training (backpropagation), the model computes error gradients from the decoder outputs, and updates network weights throughout the entire encoder–decoder model. This allows the encoder–decoder to learn using an expression-level loss function such as the cross-entropy loss between predicted output characters and their expected ground truth values.

In all encoder–decoder models for HME recognition, the attention mechanism plays a crucial role. Attention highlights salient information and helps ignore extraneous information. This mechanism addresses the limitations of the basic encoder–decoder architecture for long sequences, where information from earlier parts of the input may be lost as the algorithm processes later parts of the input. Adding an attention mechanism to the sequence-to-sequence learning model, allows important parts of the hidden state vector used in decoding to be preserved dynamically within the context vector.

Note that during decoding, the task of segmenting symbols is implicit: only characters representing symbols, spatial relationships, and region scope (e.g., {,}) are produced, but not the *input locations* for symbols, relationships, and spatial regions (contrast this with Figure 3, where these input locations are explicit). This is sometimes referred to as a 'segmentation-free' model. The decoder uses a single feature representation and a fixed set of output tokens for symbols, relationships, and spatial region scope, resulting in all four MSE recognition sub-tasks being performed in a sequential character generation model. Also note that during decoding the contextual feature vectors and attention mechanism(s) define a state guiding the output string generation, resulting in a sequential structural analysis with a graph over symbols (SRT) represented in the output string.

Zhang et al. [10] and Deng et al. [59] proposed the first encoder–decoder models that recognized math formulas from handwritten strokes and from typeset formula images (LaTeX-generated images to LaTeX strings), respectively. Le et al. proposed another encoder–decoder model using LSTM [60]. Most of these models employ CNNs and their variants such as fully convolutional neural networks or DenseNet in the encoder to extract image features. Owing to the limited availability of datasets for offline HMEs, most training images are rendered from online HME strokes, or generated using synthetic handwritten formulas. From extracted features, another deep-learning-based model (RNN [59] and BLSTM [60]) was used to encode spatial layout information. The decoder, which is an RNN [10, 59] or LSTM [60], converts the encoded features into output strings one symbol at a time. For each predicted symbol, a visual attention mechanism built into the decoder selects the most relevant region in the full HME input image, and guides the decoder to predict a mathematical symbol or an implicit spatial operator. The performance of these models generally increases with additional training samples.

Effectiveness and Popularity of Encoder–Decoder Models. The encoder–decoder models are popular for recognizing both online and offline HMEs mainly for the following three reasons:

(1) Grammar-based or graph-based methods must extract the features used in segmenters, symbol recognizers, relation classifiers, and structure analyzers. However, with the encoder–decoder model, feature extraction is part of the training process, and features for all HME recognition tasks are learned without being designed separately. Because HME patterns have complex hierarchical structures, it is difficult to determine formulations that represent these structures in a feature extractor. An encoder–decoder model has multiple layers of representation, allowing complex structures to be represented accurately when sufficient training data is available.

(2) Performance generally increases with the quality and quantity of training data, whereas it is less effective for the structural methods. For example, the performance of SCFG methods depends on the quality of the manually defined grammar rules, which cannot be automatically adapted or updated from training data. In contrast, all encoder–decoder parameters are updated simultaneously using training data.

(3) The attention mechanism in encoder–decoder models guides structural analysis automatically. In an HME, each symbol is typically related to other input symbols by the encoder. The attention mechanism combines the output of the encoder with that of decoder output symbols to create a unique context vector at each decoding time step. More generally, the attention-based encoder–decoder possesses four distinctive properties that help address limitations in conventional structure-based approaches: i. they are trainable from data; ii. the attention mechanism helps the encoder–decoder models capture syntax implicitly in weights, rather than using explicit rules (e.g., as used in grammar-based models); and iii. explicit symbol segmentation can be avoided.

3.2 Graph Neural Network Models

Graphs are more natural for structural representations than images or character sequences that represent graphs (e.g., LaTeX). Very recent works have introduced GNN-based models for HME recognition [61, 62]. These methods derive from the general encoder–decoder architecture, where an encoder produces high-level features from an input pattern, and a decoder learns a mapping between the features and an output structure. In this context, an input HME pattern is transformed into a graph of node and edge features by the encoder and then into a graph of symbols and relations (SRTs) by the decoder, enabling the use of GNNs instead of CNNs or LSTMs for more effective learning of the structures inherent in HME patterns.

Wu et al. proposed G2G, a new approach for HME recognition using GNNs with accuracy that is competitive with the state-of-the-art [61]. While encoder–decoder models treat HME recognition as a *sequence-to-sequence* problem, G2G treats HME recognition as a *graph-to-graph* problem, where the input is a graph of handwritten strokes and their relations, and the output is a graph of symbols with relations among them. The model has two main components: a feature extractor using bidirectional Gated Recurrent Units (GRU) for stroke inputs and a GNN-based encoder–decoder to construct a graph from the extracted features. Their model also uses a novel subgraph attention mechanism with graph representation learning, allowing the model to explicitly segment mathematical symbols from strokes in the input while still being trained end-to-end. This method is promising for HME recognition, as we will discuss further in Section 7.

Tang et al. introduced GETD, a GNN-based offline HME recognition model [62]. The model constructs a preliminary symbol graph from an input HME image using a symbol detector. Next, it employs a GNN-based encoder to aggregate spatial information between symbols. Then, a Transformer-based decoder identifies the symbol classes and structure from the graph to generate an output LaTeX sequence. The experiment shows that the GNN-based encoder can capture high-level features well so that GETD archives competitive results with the state-of-the-art systems (see Section 6).

3.3 Combining Neural Networks with Structural Recognition Models

While 'pure' neural network approaches have been highly effective in increasing recognition rates, structural approaches to recognition are still helpful for analysis and user interfaces. In particular, structural recognition inference mechanisms are more interpretable because the relationship between input elements and output formula interpretations is more explicit [9]. Motivated by this, neural networks have been used to adapt and improve performance for components in structural approaches.

Nguyen et al. used a combination of a max-out-based CNNs as an offline recognizer and a BLSTM as an online recognizer to improve symbol recognition on a CYK-based parser, which is similar to that shown in Figure 3 [63]. Alvaro also used a BLSTM for recognizing symbols from offline features in an SCFG-based parsing model that was the state-of-the-art model research system for its time [64]. Zhelezniakov et al. also used a BLSTM model for symbol segmentation and recognition, and

Nguyen et al. proposed a BLSTM-based model using global context to address both symbol recognition and relation classification [65]. These models benefit from information about global context captured in a deep bidirectional LSTM network, which learns temporal classification directly from online HMEs using a connectionist temporal classification loss.

Mahdavi and Zanibbi present a multi-task CNN model within a graph-based structural parsing model [66]. From a stroke-level LOS graph, visual queries are constructed for classifying stroke pairs sharing LOS edges for binary segmentation labels and spatial relationships, and to classify individual strokes as symbols. After merging strokes in symbols, a second symbol-level LOS graph is constructed and Edmonds' algorithm extracts the final symbol relation tree (SRT). This improved upon Hu's earlier LOS-based model [42], but the design of the two recognition stages (e.g., merging detected symbols from independent classifications of strokes pairs) uses less shared contextual information than 'pure' DNN models, resulting in lower accuracy.

4 Advances in Encoder–Decoder Models

The principal challenge of the encoder–decoder models is improving their generalization, which often requires a large quantity of training data. Moreover, training the models is difficult because of the structural complexity of HMEs. For long and complex expressions, in encoder–decoder models it can be difficult to determine where the model should attend at each decoding step. In this section, we present advances in the encoder–decoder models that improved their performance.

4.1 Additional Constraints for Training, Encoding, and Decoding

Many studies [10, 11, 57, 67] have used not only an expression-level loss but also auxiliary losses to improve the decoding step of the models. Zhang et al. proposed input coverage attention to support the training process of their Watch, Attend, and Parse (WAP) models [10]. Truong et al. improved their encoder by applying weakly supervised learning (WSL) to recognize handwritten symbols, termed Rec-wsl [57]. Li et al. used the number of occurrences for each symbol class as a constraint to improve training of the WAP model [68]. Nguyen et al. improved the encoder using a connectionist temporal classification loss [67]. Li et al. proposed a module to distinguish similar symbol classes [69], where a path signature feature [70] and a language model reflecting contextual information were used to disambiguate visually similar symbols, while a method based on Dynamic Time Warping (DTW) constrained the alignment of symbols within the HME input. Recently, Guo et al. proposed PrimCLR, a two-stage training procedure [71]. First, contrastive constraints and unsupervised learning are used to pretrain the encoder and decoder components using patterns from CROHME-Hybrid [72] and HME100K datasets [73]. Second, the pre-trained components are adapted for downstream HME recognition using supervised fine-tuning. Liu et al. used a co-occurrence matrix to represent the relationship between symbols in an HME and proposed a semantic aware module to enhance the encoder–decoder model by learning the correlation between different symbols [74].

Unlike the aforementioned methods, which are applied to offline images, Zhang et al. proposed the Track, Attend, and Parse (TAP) architecture that parses an online HME into a LaTeX sequence by tracking a sequence of input points [11]. Initially, trajectory information was extracted from the sequential points of the input pattern. The encoder or tracker stacks several layers of bidirectional GRUs to obtain a high-level representation. To support the training of the attention model, TAP uses symbol-level annotations as constraints to help the model learn where it should attend during decoding.

4.2 Multimodal: Combining Online and Offline Inputs

In contrast to other encoder–decoder models trained using handwritten strokes for HMEs with LaTeX strings as ground truth annotations, many studies use multiple input modalities to train their models [11, 58, 75], combining handwritten strokes with formula images (e.g., generated from the strokes). Figure 5 illustrates an example of such a model. This approach is motivated by handwritten strokes and images containing different types of information. Online handwriting includes temporal information and makes the strokes used to draw symbols explicit, whereas offline handwriting captures the visual information of an input pattern without access to strokes or temporal information.

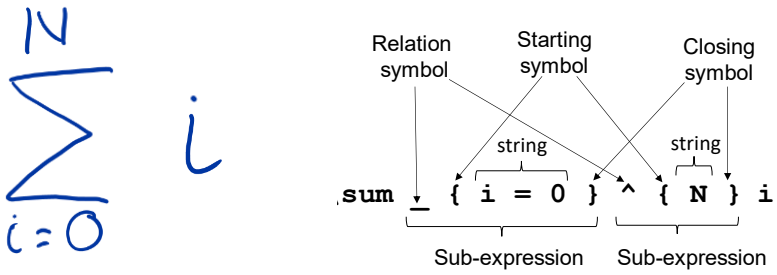
Wu et al. proposed a scalable paired adversarial learning model trained using a dataset of real HME images with printed templates (LaTeX- generated images [12, 58]). This novel approach learns semantic-invariant features for offline HME recognition that can handle complex two-dimensional structures and various writing styles. Its effectiveness was demonstrated by achieving the top accuracy amongst participating systems in the CROHME 2019 competition. Similarly, Le proposed a dual-loss attention model that utilizes an existing LaTeX corpus to improve accuracy for HME recognition [75]. Their model aims to learn semantic-invariant features between handwritten and printed mathematical expressions for the encoder and LaTeX grammar for the decoder from handwritten and printed mathematical expressions.

Wang et al. proposed the MAN model [76] that combines online [11] and offline models [10]. The combined model can be trained end-to-end by using a DNN. Similarly, Wang et al. proposed SCAN that combined online and offline HME recognition models [77]. The hybrid model captures both the temporal (online) and visual (offline) information from input formulas. Experiments show that hybrid models perform better than models using a single modality, as we will discuss further in Section 7. Furthermore, multimodal training data are now available: CROHME 2019 [78] and CROHME 2023 [79] provide both online and offline modalities of the same expressions. The offline samples can be generated from the online signal (as in CROHME 2019) or scanned from the original image (as in CROHME 2023).

4.3 Transformer-Based Encoder–Decoder

Transformers rather than RNNs have been used as decoders for HME recognition. Ding et al. used a Transformer to improve an RNN-based decoder [80]. Zhao et al. recently proposed BTTR, a Transformer-based HME recognition model [81]. Moreover, the bidirectional training strategy helps the Transformer-based decoder learn to decode from both the forward and backward directions of a LaTeX sequence, which further increases the recognition rate. However, similar to general encoder–decoder models, the models parse HME structures without taking math expression syntax explicitly into account.

Attention coverage can be applied to Transformers. Recently, Bian et al. improved the BTTR model using an attention aggregation module to integrate multi-scale coverage attentions into the BTTR model [82]. Zhao et al. proposed an Attention Refinement Module based on the coverage attention mechanism to refine attention in the Transformer decoder [83]. Specifically, the module generates current attention by coverage attentions with different scales during decoding. Lin et al. used a combination of contrastive learning and supervised learning for training the BTTR model [84]. The proposed method can help address the lack of HME patterns while training the model. Similarly, Wang et al. used an asymmetric Siamese network that narrows the gap between HME images and the printed templates for better feature extraction [85].



(a) HME pattern for "$\sum_{i=0}^{N} i$." (b) LaTeX representation for "$\sum_{i=0}^{N} i$."

Fig. 6. Example of a summation expression with 2D structure.

4.4 Limitations in Sequential Output Representations and Language Models

In general, current DNN-based models do not use context effectively to identify invalid output expressions. Truong et al. showed that a one-dimensional LaTeX sequence is not effective for representing the two-dimensional structures of mathematical expressions [86].

A LaTeX sequence represents the two-dimensional structure of an HME as a one-dimensional sequence and uses a pair of symbols `{` and `}` to define the mathematical scope for each subexpression, as shown in Figure 6. Note that nested subexpressions in LaTeX (e.g., in an exponent) contain not only visible symbols, but also associated spatial relationship symbol, and the two 'scope' symbols `{` and `}` which *do not* exist in the input. The need to identify subexpression scopes without explicit input features can make symbol segmentation and classification ambiguous. A standard encoder–decoder model generates a LaTeX representation without considering this ambiguity, and so a decoder may generate an ungrammatical sequence such as `x^{2` and `x^{2}}` for the expression `x^{2}`. To overcome this problem, many systems use linguistic context from mathematical language models (LMs) to support the decoding process [11, 58, 87, 88]. While these LMs tend to be much simpler than full expression grammars, they serve the same purpose by introducing syntactic constraints in the space of possible expressions.

Language Models (LMs) have often been used to resolve ambiguities in HME recognition [20, 11, 19, 58, 87, 88], although they are generally not as effective as for natural languages where redundancy is much higher, and so misrecognized characters can often be replaced reliably by correct characters. However, mathematical expressions follow a more formal language with less redundancy (e.g., a specific superscript in a formula appears just once). Nevertheless, they still follow grammatical rules, and semantic constraints with varying likelihoods; and LMs are often a last resort to resolve ambiguities in neural HME models. A merit of symbolic LMs is that they can be trained using a large set of mathematical expressions without requiring handwritten expressions (e.g., when trained using a large corpus of LaTeX formulas).

For structural approaches, LMs have been added to capture additional linguistic context, often using N-grams. For example, the commercial systems of Myscript and Samsung utilize LMs to resolve ambiguities [20, 19]. Zhelezniakov et al. published their method for the latter system, where one bi-gram LM was used to model language sequences, and another bi-gram LM was used to model language relations for expression construction [19].

For DNN approaches, Zhang et al. used a GRU-based LM in addition to their HME recognition model [11]. Similarly, Wu et al. [58] and Truong et al. [87] incorporated an N-gram LM to support the decoder of their HME recognition models. All methods used a large number of LaTeX strings to train the LMs.

Recently, Ung et al. proposed the use of a Transformer-based LM to capture context in LaTeX sequences within mathematic formulas [88]. Based on the self-attention mechanism of Transformer Networks, the high-level representation of an input token in a sequence of tokens is computed by how it is related to the previous tokens. Thus, the Transformer-based LM can capture long -range dependencies and correlations between symbols and relationships in a math formula.

Tree-Structure Decoders and Grammar Rules. Other studies also use graphs or SRTs as the output representation instead of LaTeX [15, 89, 90, 91]. Zhang et al. proposed a tree-based BLSTM [15] that directly produces a graph describing an HME. Zhang et al. proposed tree-structured encoder–decoder models based on a sequential relation decoder (SRD) for offline [89] and online HME recognition [90]. Both models attempt to consider grammatical information by decomposing the target SRT into a subtree sequence, where each subtree has a parent-child relationship. Although a tree-structured decoder model generally exhibits greater robustness than a LaTeX-based decoder, existing tree-structured decoders generate an SRT tree without considering syntactic constraints.

Further improvements to the tree-structured encoder–decoder have been made recently. Wu et al. proposed TDv2 for offline HME recognition [92], which is free from the relation order of the input. They also added symbol-level and pixel-level auxiliary constraints while training their model. Wang et al. proposed the Memory Relation Decoder (MRD), equipped with a memory-based attention model to improve accuracy when determining the next symbol node [91].

Yuan et al. proposed a Syntax-Aware Network (SAN) equipped with grammar rules that efficiently divides a syntax tree into different components to alleviate errors caused by tree-structured ambiguities [73]. The SAN integrates syntactic constraints into the parsing step of the encoder–decoder model to form a parse tree. It learns to generate grammatical relationships and navigates a parse tree to create components based on these relationships.

4.5 Ensembles

Several studies have adopted the ensemble method [93] to reduce overfitting and improve performance [10, 11, 57, 90]. They trained several encoder–decoder models with different initial states. Each model can generate different output candidates for the input HME pattern, and the best candidate was chosen based on the highest average score of the predicted symbol during the beam search process.

5 Data Augmentation and Generation

Data augmentation and generation are used to expand available training data. Here, data augmentation refers to increasing the number of samples by applying simple distortions to the existing samples, whereas data generation refers to the generation of new samples. We start with a brief review on data augmentation, and then discuss data generation. Data augmentation is widely used in modern machine learning systems, including for HME recognition.

5.1 Data Augmentation Methods

Data augmentation has been successfully applied in several fields including object detection [94], handwritten text recognition [95, 96], and HME recognition [60, 97, 72]. In most cases, data augmentation uses affine transformations (i.e., translation, rotation, scaling, and shearing) to generate samples that vary the shape and position of available samples. DNN models are able to learn features that are invariant to these transformations from large datasets of augmented samples.

Le et al. used a pipeline of local and global distortions to augment HME formula data [72]. Local distortions affect symbol patterns, whereas global distortions affect the whole expression. However, they did not show substantial improvement because HMEs include not only information regarding symbols but also information regarding the spatial relations among them. Their augmentation method is unable to vary spatial relationships.

Li et al. proposed a method to augment samples at different scales [97]. Their proposed method improved the performance of an HME recognition model, producing a large improvement in accuracy for the CROHME dataset. However, it only addresses the scaling problem of the HMEs. Adding additional global and local distortions for symbols inside HMEs in this model may be beneficial.

5.2 Data Generation Methods

The goal of data generation is to generate diverse but syntactically valid handwritten formula training samples. For example, Graves used RNNs to generate handwritten samples from text sequences [98]. Alonso et al. used a generative adversarial network to generate handwritten text patterns [99].

The two-dimensional structures in HMEs are less constrained by linguistic context, whereas texts are one-dimensional and are strongly constrained by linguistic context due to differences in the level of redundancy in text vs. formulas. This makes HME generation more challenging than both text and handwritten text generation. Le et al. applied decomposition to extract sub-HMEs as new training samples [72]. They treated an online HME as a combination of sub-HMEs, using grammar rules to decompose an HME into sub-HMEs, and then used the subexpressions as new formulas with notable improvement. However, their method is still limited to the supply of complete formulas, because local structures are combined using fixed templates.

MacLean et al. used a mathematical grammar to automatically generate a large number of mathematical expressions [100]. The grammar constrains generated LaTeX to follow grammatical rules, which validates the generated formulas. However, HMEs are then manually written by many people for the generated notations; therefore, it seems labor-intensive to prepare a large number of HME patterns as pointed out by Deng et al. [59]. Then, they proposed a method for synthesizing HME patterns from LaTeX strings, and collected handwritten symbol patterns. Their method generates LaTeX strings and then replaces all individual symbols with handwritten symbols without applying structural and stylistic variations. Similarly, Khuong et al. presented a method for generating realistic HMEs with a wide variety of structures and styles from a single LaTeX sequence [101]. They created a template that uses a symbol-relation tree constructed from an input LaTeX or MathML string, replacing symbols in the template with handwritten symbols.

These methods [100, 101] generate a large number of HME patterns by replacing symbols in generated templates. However, they only generated HME patterns from given LaTeX sequences. Furthermore, their methods use fixed templates to construct HMEs, which differs from how humans write mathematical formulas. Training samples with invariant writing styles cause the HME recognition methods to overfit the data and do not work well for real HME patterns.

Recently, Truong et al. proposed a method to generate many syntactic HME patterns [87]. This approach has been used in CROHME 2023 [79] to generate 76,224 new training samples in the competition. They used a grammar to decompose expressions using syntactic rules, and generate new structures/patterns by subexpressions interchange. The method also uses local distortions for the substructure and global distortions for the whole HME pattern during the generation process. The generated patterns are used to train not only the HME recognition models but also the mathematical LMs. Yang et al. proposed

a tree-based data augmentation and generation [102]. Their method supports tree-based operations to generate new patterns.

Table 2: Public datasets and their properties.

Data set	Type	Amount	Format
MathBrush [127]	Expression	Total: 4,655	Microsoft and SCG ink
CIEL [129]	Symbol	34 classes	MathML
	Expression	Train: 6,480 Test: 3,600	
HAMEX [126]	Expression	Train: 2,925 Test: 1,425	MathML
MfrDB [128]	Expression	Total: 2,018	MathML
ExpressMatch [125]	Expression	Total: 910	MathML
CROHME 2011 [4]	Symbol	56 classes	MathML
	Expression	Train: 921 Test: 348	
CROHME 2012 [103]	Symbol	75 classes	MathML
	Expression	Train: 1,341 Test: 486	
CROHME 2013 [104]	Symbol	101 classes	MathML
	Expression	Train: 8,836 Test: 671	
CROHME 2014 [20]	Symbol	101 classes	MathML
	Expression	Train: 8,836 (train CROHME 2013) Test: 986	
	Matrix	Train: 362 Test: 175	
CROHME 2016 [105]	Symbol	101 classes	MathML
	Expression	Train: 8,836 (Train CROHME 2013) Validate: 986 (Test CROHME 2014) Test: 1,147	
	Matrix	Train: 362 (Train CROHME 2014) Validation: 175 (Test CROHME 2014) Test: 250	
CROHME 2019 [78]	Symbol	101 classes	MathML
	Expression	Train: 9,993 (Train CROHME 2016 + Test CROHME 2013 + Test CROHME 2014) Validate: 986 (Test CROHME 2014) Test: 1,199	MathML and rendered images
OffRaSHME [124]	Symbol	102 classes	Scanned images
	Expression	Train: 19,749 Test: 2,000	
HME100k [73]	Expression	Train: 74,502 Test: 24,607	Scanned images
CROHME 2023 [79]	Symbol	101 classes	MathML and Scanned images (Bimodal)
	Expression	Train: 19,979 (+ 145,108 artificial patterns) Validate: 1,702 Test: 2,300	

These methods [87, 102] are based on the pattern augmentation and generation used in a previous study [72].

6 EVALUATION OF RECOGNITION METHODS

In the last decade, many datasets have been collected and made publicly available to the community. In parallel, a number of evaluation metrics have been proposed and used to establish benchmarks using the publicly available data. We described these, along with the performance of various HME recognition models on these benchmarks in the remainder of this Section.

6.1 Public Datasets

Public datasets are useful for benchmarking the performance of pattern recognition systems, although they require considerable human effort to create. The formation of a public dataset for HME recognition includes two main tasks: data collection and the creation of ground truth. Data collection can be performed in two ways: copy style, where the writer is given mathematical formulas and must copy them, and free style, where the writer is free to write formulas as they like. Formulas collected using the free style of data collection may result in more natural handwritten patterns with realistic variations and distortions but requires much more human effort to define ground truth. Dataset editors must analyze handwritten patterns and annotate them based on their locations, labels, layouts, and content. MacLean et al. proposed a grammar-based technique to assist with the creation of ground truth for copy style data collection [100].

Table 2 lists available public datasets for HMEs as well as their formats and properties. In 2011, CROHME was introduced, which later became the standard for evaluating HME recognition systems [20, 4, 103, 104, 105]. All datasets in Table 2 have online inputs (i.e., handwritten strokes), except for OffRaSHME and HME100K which provide scanned images of HMEs. CROHME 2019 provides both online and rendered offline HME patterns. Offline patterns can be easily generated from online patterns, which are commonly used in offline HME recognition [10, 12, 57]. However, it should be noted that there are two main differences between the actual offline images and simulated offline images.

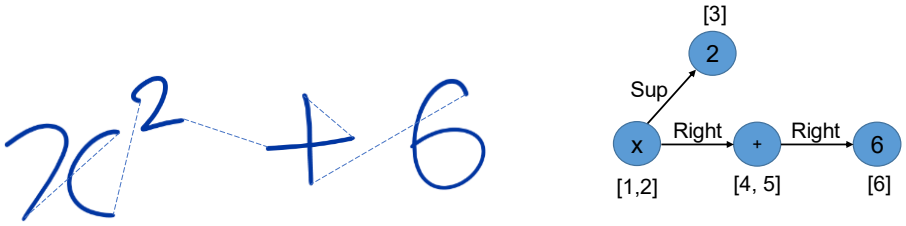
- Real offline images typically contain uneven stroke widths, grayscale, blurs, noise, and sometimes showthroughs.
- Writing with pen on paper is different from writing with stylus on a digital device, which can alter writing styles.

Recently, CROHME 2023 [79] provides both online and offline modalities for identical expressions. This newfound availability of multimodal datasets presents a promising solution for training hybrid models.

6.2 Evaluation Metrics

During the last decade, several evaluation methods focusing on math formula structure have been proposed. Sain et al. proposed the EMERS metric [106], which is a graph edit distance to compare output SRTs with a ground-truth tree. The number of changes required to convert the recognized tree into the ground-truth tree is computed as the distance between trees. However, due to lack of well-defined tree structure which could enforce uniqueness in representing mathematical formulas, the comparison of the recognized tree and ground truth tree is biased to the tree structure. Alvaro et al. proposed a parser to generate a new tree (t_P) from the ground truth tree (t_G), which is semantically equivalent to the ground-truth tree (t_G) and syntactically equivalent to the recognized tree (t_R). Then, they counted the number of errors by comparing t_R with t_P using the EMERS metric [107].

Zanibbi et al. proposed a different online evaluation scheme based on input strokes [108] and labeled directed graphs. Distances are computed using hamming distances over labeled graphs representing segmentation, classification, and spatial relationship decisions at the input stroke level. Error is defined by the number of node and stroke-pair edge relabelings needed



(a) HME pattern for "$x^2 + 6$." (b) SLT for "$x^2 + 6$."

Fig. 7. Example of expression with its SLT. Stroke indices are shown in the brackets.

to match the ground truth expression. This approach avoids the need to align SRTs with different numbers of symbols and/or relationships, and captures all segmentation, classification, and relationship errors using the input strokes and stroke pairs. This evaluation model was developed and later modified for the CROHME competitions (starting with CROHME 2013), where originally ground truth included which strokes were included in symbols and relationships in the target formula.

The CROHME competition was introduced in 2011 by Mouchère et al. to provide a standard dataset and tools for automatic evaluation of online HME recognition systems. In all CROHME competitions, systems are ranked by expression rate. From CROHME 2011 to CROHME 2016 [20, 4, 103, 104, 105], the evaluation metrics were revised to enable researchers to analyze errors in greater detail, including a tool to automatically compile and sort recognition errors for symbols and SRT subgraphs by frequency (using the *confHist* tool). Figure 7 illustrates the example of an HME pattern of $x^2 + 6$ and the corresponding SRT with stroke annotations. In the SRT, a set of labeled strokes (nodes) defines a symbol, and a labeled directed edge between strokes defines a spatial relationship. Metrics at the level of strokes, symbols, and expressions were used to evaluate systems. (including for matrices).

While the original CROHME evaluation was able to identify and compile errors automatically, it was assumed that parsers were structural approaches producing SRTs with stroke annotations as shown in Figure 7 (e.g., using Presentation MathML annotated with stroke data, or equivalent labeled graph (lg) representations). Unfortunately, encoder–decoder models are 'segmentation-free', and generate LaTeX strings without reference to input strokes. Directly comparing recognized LaTeX strings to ground truth LaTeX strings is not a good strategy for evaluation: in LaTeX, a single formula can be represented by multiple LaTeX strings. For example, "x^2_1," "x_1^2," and "x_{1}^{2}" all represent the formula x_1^2.

One possible solution for this many-to-one relationship between LaTeX strings and formulas is to constrain the LaTeX sequences for each mathematical formula (i.e., normalize syntax for LaTeX string outputs). Using a canonicalized LaTeX representation also helps encoder–decoder models learn through constraining the output space. However, this method requires ground truth data to be in this constrained representation. Another possibility is the IMEGE metric proposed by Alvaro et al. [107], which compares the rendered LaTeX representation to that for ground truth to measure accuracy. This evaluation method resolves a number of problems related to non-canonical LaTeX strings, but can be sensitive to differences in symbol sizes and visual artifacts.

Since CROHME 2019 new metrics were created by adapting the existing label graph-based evaluation framework, to both address the LaTeX canonicalization problem, and to allow stroke-based label graph outputs and LaTeX strings to be compared at the *symbol* level after converting LaTeX strings and label graphs to SRTs (in MathML format). While this provides a method for detailed evaluation of encoder–decoder systems, it also creates difficulties when comparing newer systems to results from previous CROHMEs. Indeed, in CROHME 2014, a valid expression has a correct segmentation and identification of each symbol

and relation, but since CROHME 2019, the link with raw input is optional, and not considered unless the original stroke-based label graph evaluation model is used.

Finally, the above-mentioned methods require ground-truth scripts that are often manually generated by humans. Tools that partially or fully automate ground truth creation in conjunction with the augmentation and generation methods described in Section 5 would bring many benefits, both for evaluation and HME data collection.

Table 3: ExpRates on CROHME testing sets (%).

Approach		On. HME	Off. HME	Testing sets / Systems	CROHME 2014 testing set	CROHME 2016 testing set	CROHME 2019 testing set
Structural	Grammar	✓		MyScript [20, 105, 78]	62.68†‡ (+30,000 HMEs)	67.65†‡	79.15†‡
		✓		Samsung R&D [78, 19]	-	65.76†	79.82†
		✓		València [20] /Wiris [105] /	37.22	49.61	-
		✓		TUAT [20, 105, 78]	25.66	43.94	39.95
	Graph	✓		Sao Paolo [20, 105]	15.01	33.39	-
		✓		Nantes [105]	26.06	13.34	-
		✓		Tree-BLSTM [15]	29.91	27.03	-
		✓		Tree-construction [44]	44.12	41.76	-
DNN	Encoder–Decoder(ED)	✓		MathType [78]	-	-	60.13†
	ED + Constraint		✓	WAP* [10]	44.40	44.55	-
			✓	Rec-wsl* [87]	55.68	52.57	53.46
			✓	DenseMSA* [123]	52.80	50.10	41.70 (w/o ensemble)
			✓	CAN-ABM [68]	57.62	56.15	55.96
		✓		TAP [11]	48.50	44.80	
	ED + Multimodal	✓	✓	WAP*&TAP*&LM (+ 173,500 LaTeX seq.)	61.16‡	57.02‡	-
		✓	✓	MAN* [76]	54.05	50.86	
		✓	✓	SCAN* [77]	57.20	53.97	56.21
	ED + Transformer		✓	BTTR [81]	54.00	52.30	53.00
			✓	CCLSL [84]	58.07	55.88	59.63
			✓	CoMER [83]	59.33	59.81	62.97
	ED + Tree-based representation		✓	DenseWAP-TD* [89]	54.00	52.10	54.60
			✓	TDv2 [92]	53.62	55.18	58.72
			✓	SRD* [90]	53.60	50.40	50.60
			✓	MRD [91]	55.80	52.50	53.60
			✓	SAN* [73]	61.30	61.50	62.10
	ED + Data aug. & gen.		✓	Tree-based gen. + DWAP* [102]	61.63	59.81	64.38
			✓	Syntactic gen. + Rec-wsl* [87]	57.20	59.20	56.13
	ED + Composition of methods		✓	PAL* [78, 58]	47.06	-	71.23† (+120,000 HMEs)
			✓	PAL-v2*&LM [78, 58]	54.87	57.89	62.89
		✓	✓	USTC-iFLYTEK (online) [78] based on WAP*&TAP*&LM	-	-	80.73‡ (+590,000 LaTeX seq.)
			✓	USTC-iFLYTEK (offline) [78] based on WAP*&LM	-	-	77.15‡ (+590,000 LaTeX seq.)
			✓	Rec-wsl* & LM [87]	64.60	66.08	58.72
	ED (GNN)	✓		G2G [61]	54.46	52.05	
			✓	GETD [62]	53.45	55.27	54.13

"∗" denotes an ensemble of several differently initialized recognition models.
"&LM" denotes using a mathematical LM.
"†" denotes using additional private HMEs.
"‡" denotes using additional LaTeX sequences for training LM.

6.3 Evaluation on Public Datasets

In this section, we present an in-depth analysis of the CROHME and OffRaSHME competition results, particularly focusing on CROHME 2019, which yields insights into the successes and challenges [5, 78, 105] encountered in both online and offline HME recognition.

Table 3 categorizes the HME recognition methods as structural and DNN approaches, with additional sub-classifications. Recognition rates on the CROHME 2014, 2016, and 2019 testing sets are provided[1]. It also shows whether each method is for online and/or offline HME patterns, uses a model ensemble, a language model (LM), additional private HMEs for training, or additional LaTeX sequences for training the LM.

Table 4 presents the recognition rates of the HME recognition methods on the OffRaSHME testing set for offline HME recognition. It also shows recognition rates when one ($\leqslant$ 1 symbol errors) to two ($\leqslant$ 2 symbol errors) errors in the SRT are permitted, giving a sense of the robustness for each system. In addition, the table lists the structural recognition rates of different systems while ignoring the symbol category. All the listed methods used the provided dataset without any additional HME patterns to train their models. It also shows whether each method avails the model ensemble and the LM.

Although several other methods claim higher accuracies compared to the listed methods that have participated in CROHME competitions, most report results using data that is not publicly available. Therefore, the results of these methods remain difficult to reproduce, and we cannot evaluate their true potential based on their reported accuracies. Even among the methods listed in Table 3, conducting a completely fair comparison is difficult due to the use of additional training patterns in some cases. Nevertheless, the performance on the CROHME dataset provides reliable insights into the recognition methods when the conditions for the recognition methods are clearly stated.

For structural approaches, grammar-based methods achieve higher ExpRates than graph-based methods. MyScript achieved the highest ExpRate in the CROHME 2014 and 2016 competitions, although it should be noted that it is trained by 30,000 additional HME patterns and its LM is trained by additional LaTeX sequences [105]. The top three methods in CROHME 2016 (Myscript, Wiris, and TUAT) used the CYK algorithm and incorporated heuristics into the methods via a set of high-quality grammar rules. This is consistent with the idea that grammar-based methods can restrict interpretations more effectively than graph-based structural methods when grammar rules for mathematical expressions are carefully designed.

For decades, structural approaches were the standard family of techniques for HME recognition. However, grammar-, tree-, and graph-based methods [14, 42, 109] have demonstrated performance limitations resulting from their dependency on predefined grammar rules and handling subtasks for HME partially or wholly independently, limiting the use of contextual

Table 4: ExpRates on OffRasHME testing sets (%).

Rank	Systems	ExpRate	≤ 1 symbol errors	≤ 2 symbol errors	Structure
1	USTC-iFLYTEK* [124]	79.85	89.85	92.00	92.55
2	SCUT-DLVCLab [124]	72.90	86.05	88.80	89.45
3	TUAT(Rec-wsl*&LM) [124]	71.75	82.25	85.80	86.60
4	HCMUS-I86 [124]	66.95	79.65	83.60	84.00
5	SYSU [124]	61.35	77.30	81.55	82.90

"$*$" denotes an ensemble of five differently initialized recognition models.
"&LM" denotes using a mathematical LM.
"$\leqslant$ *n* symbol errors" denotes ExpRate when *n* symbol errors in the SLG are permitted.

[1]Recognition rates on CROHME 2023 was not included since the rates for it are not available for most of the methods.

information. As discussed above, a number of structural approaches have made use of DNNs for symbol segmentation, symbol classification, and relation classification [19, 65].

In the past decade, encoder–decoder models with attention mechanisms [10, 11, 12, 57] demonstrated their effectiveness for HME recognition. Because subtasks are handled jointly using a shared context, the DNN approach outperforms the structural approach. On the CROHME 2014 testing set, WAP, the first offline encoder–decoder model, recorded an expression rate of 44.4%, which was higher than the other structural methods (TUAT, Sao Paolo, Nantes). TAP, the first online DNN-based HME recognition model, achieves a 48.5% of ExpRate on CROHME 2014.

As discussed earlier, advances in encoder–decoder models have included adding constraints, using hybrid models, Transformer decoders, and tree structure-based decoders. Adding more constraints improved the ExpRate of the WAP model to 55.68% (Rec-wsl), 52.8% (DenseMSA), and 57.62% (CAN-ABM) on CROHME 2014. USTC-iFLYTEK used an encoder–decoder model trained with additional LaTeX sequences and generated HMEs from the provided training set to achieve the highest expression rates of 80.73% (Table 3) and 79.85% (Table 4) on CROHME 2014. Hybrid models combining online and offline HME recognition models (MAN, SCAN, WAP*&TAP*, and USTC-iFLYTEK) have achieved higher ExpRates than those using only online or offline inputs.

Transformer-based decoders can be used instead of RNN-based decoders for HME recognition, which include self-attention and multi-head attention that can capture long-term dependencies. BTTR, CCLSL, and CoMER yielded comparable ExpRates to the other DNN methods. CoMER achieved ExpRates of approximately 60%, which is comparable with the top methods for all the CROHME testing sets.

Tree-based encoder–decoder models have been proposed to use SRTs as the output representation, which is more natural than LaTeX strings for representing two-dimensional structures in mathematical expressions, and can capture additional syntactic information. The SAN model integrated the tree-based encoder–decoder model with a grammar, producing promising results. DenseWAP-TD, SRD, TDv2, and MRD achieved ExpRates of more than 50% on the CROHME testing sets.

By adding grammar-driven parsing, the SAN increased ExpRates for tree-based encoder–decoder models to more than 60% on the CROHME testing sets. The models that use tree structure-based decoders, Transformer decoders, and GNN-based encoder–decoder achieved promising performance without using additional HMEs and LaTeX sequences for training the recognition model and LM, respectively.

On the other hand, the GNN-based encoder–decoder models, G2G and GETD, achieves comparable ExpRates with other DNN-based methods on the CROHME 2014 and 2016 testing sets. Even the DNN approaches cannot completely solve ambiguous cases in the context and semantics of input HMEs, which remains a challenge for HME recognition.

In Table 3, the top systems MyScript, Samsung, and MathType use additional HME patterns for training the recognition model to achieve high expression rates (ExpRates). Additionally, USTC-iFLYTEK and MyScript use additional LaTeX sequences for training the LM . USTC-iFLYTEK and Rec-wsl show that generating HME extra data from a given training set helps to improve the ExpRate. These results underscore the importance of data augmentation and generation in improving performance. The ensemble architectures marked by "*" in Table 3 achieved the best performance for each method, as stated in the cited papers. An LM was also incorporated to improve the expression rate (ExpRate) for each of the top models of CROHME 2019 (USTC-iFLYTEK, PAL-v2, and Rec-wsl).

The methods shown in Table 4 exhibit similar performance because they have been published more recently, and they use many techniques already invented for online patterns. The DNN approach for offline HME recognition produced a performance comparable to that of online recognition without tailoring segmentation, symbol classification, relation classification, and structural analysis, which has rekindled interest in offline HME recognition.

7 PROGRESS AND REMAINING CHALLENGES

This section summarizes progress to-date, along with remaining issues that may be the focus of future work. In the last decade, there has been a marked improvement in the recognition rates of HMEs. Starting from expression rates of 20% to 60% of in CROHME 2014 using structural approach, this has been improved to between 50% and 80 % owing to the shift to neural recognition models (especially encoder–decoder models), ensemble models, LMs, hybrid models, and data augmentation based on the public HME datasets. The advances in neural models and data augmentation/generation described above in Sections 4 and 5 are largely responsible for this dramatic improvement in recognition accuracy. These models are also generally easier to design and implement than the earlier structural approach models.

For the subtasks of HME recognition, over 98% correct segmentation [20], 95% symbol recognition, and 88% structure recognition [105] rates have been achieved for CROHME 2016, even in the middle of the last decade. Obtaining high performance in the sub-tasks of structural recognition are no longer difficult challenges, but there remains room for improvement.

7.1 Remaining Challenges

The recognition of complete handwritten mathematical expressions still shows a performance of slightly over 80% for online recognition and nearly 80% for offline recognition [105]. This recognition rate likely degrades as the number of symbols and relationships in a formula increase.

Local and global relationships among neighboring and distant symbols in an HME are able to provide contextual information that complements constraints from grammars for mathematical expressions, but they have not yet been explored in a systematic manner. Even the latest DNN-based methods learn to parse HME structures using only spatial relations without syntactic context [7, 8, 57]. Grammars using for HME recognition (whether graph or string) are context-free. Context used within encoder–decoder models so far is linear in the LaTeX representation, or it is within each subsequence in a tree, such as a path from the root to each leaf, even in tree representations. More context must be utilized in HME recognition to better cope with ambiguities in symbol segmentation, symbol classification, relation classification, and structural analysis.

Matrices have been recognized only by a few systems, probably because they are uncommon in the public databases. Complex mathematical expressions such as those used in Physics are also left. Although some methods to recognize tables or structures might be better combined, we won't expect significant difficulties in recognizing them. In any case, it must be proved in future studies.

Data is important for the DNN approach. More public datasets are essential to encourage research and development of HME recognition. Moreover, hybrid datasets of online and offline samples from the same HMEs are useful. Related to this, data collection methods are a key remaining challenge. Often, recognizers trained by copied HME patterns fail to recognize freely written HMEs. To make HME recognizers more useful for real applications, they must be trained using freely written formulas. To collect free-style HME patterns, the HME patterns should be collected from real applications, such as Computer Based

Testing. However, preparing samples in the free style is costly because their ground truth annotations must be provided for each sample. Nevertheless, such datasets will be an important milestone in HME recognition.

There also remain challenges with turning research on HME recognition into practice on a large scale. First, the majority of users who input HMEs on pen and touch devices are children and young students, but their HMEs are very different from the high-level HMEs provided by CROHME. Second, HME recognition is useful not only for anonymous users, but also for specific daily users, and so user customization and adaptation are required to avoid frustrating users. Third, good user interfaces with real-time recognition feedback [48] are required, and associated challenges in DNN approaches have not yet been addressed [9].

8 Related Research Topics

So far, we have been discussing single-line HME recognition. In practice, however, often multiple lines of HMEs need to be recognized. A competition on multi-line HME recognition was held in ICDAR 2023 [110]. Like multi-line text recognition, the component of HME line segmentation is added. Context analysis among multi-line HMEs is a challenging task. Another extension is to recognize a mixture of natural language text and HMEs. Some previous works were made for printed text and printed mathematical expressions [111], but not yet seriously for handwritten text and math. The National Center for University Entrance Examinations in Japan collected offline handwritten answers in Japanese language and mathematics from approximately 120,000 people in trial tests conducted in 2017 and 2018 for investigating to include descriptive answers in the university common entrance examination (UCEE). Approximately 500,000 examinees nationwide take the UCEE exam annually. Multiple-choice questions have been answered in a mark sheet and scored automatically for many years, but handwritten descriptive answers were considered to be included to test thinking ability more straightforwardly. Handwritten answers in mathematics include multi-line HMEs and Japanese text, so that their recognition should be vital for automatic scoring.

Automatic scoring of handwritten answers is a natural extension of handwriting recognition research and it is composed of handwriting recognition, automatic scoring, and natural language understanding. The performance of automatic scoring of constructed response answers in Japanese language for the trial tests were reported [112, 113], the study of this for mathematics answers just started. It has been long sought, since tablets were introduced for learning [114, 115, 116]. ASSISTments [117] develops an intelligent learning platform for students. It focus on math problems and required photos of the students' answers so that the human or computer-based scorers could give a score for each answer. Recent advances in HME recognition may drive the automatic scoring research in the next generation of educational assessment systems [118].

HME clustering is also an important research topic. HME recognition can be used for automatic scoring of HME answers, whereas HME clustering is used for examiners to mark them efficiently and reliably, placing HMEs with similar patterns into the same group. This requires the extraction of features from HMEs and produces a similarity between two HMEs. Khuong et al. adopted a structural approach and proposed multilevel features to represent offline HMEs [119], and suggested that the time for scoring HME answers is reduced when using automatic scoring rather than manual scoring. Ung et al. used the same approach and proposed HME clustering for online HMEs [120]. Recently, Nguyen et al. proposed a CNN-based model for learning both class and location representations of symbols for clustering online HMEs [121]. The problem remains to collect more real HMEs and demonstrate the effect of HME clustering.

In real word use cases, mathematical formulas are not isolated. One direction for progress for HME recognition is to better take into account the context of each formula. Some contexts are common with printed expression recognition task: included in a sequence during a demonstration, between two paragraphs describing a concept, or embedding in a sentence in plain text. HME

recognition can use more complex context because of different usages: audio recordings for scientific lectures (as in [122]), free notes or schemes around the equation. These contexts can provide additional information to improve recognition. The consistency of variable usage is an obvious solution to solve ambiguities, but creating the link between entities is not simple.

Inferring the mathematical domain is also a way to guide the grammar and the vocabulary of the HMEs to recognize. In these examples of context driven HME recognition, dedicated data sets are not yet available, but recognizer architectures based on encoder–decoder are ready to include this contextual information. Recent advances in the other related domains (speech recognition, Natural Language Processing, Video analysis, …) will allow us to address these complex contextual recognition tasks in future years.

9 CONCLUSIONS

Machine recognition of Handwritten Mathematical Expressions (HMEs) may be understood as a specific type of multidimensional text recognition, where symbols in formulas are organized on writing lines that are laid out hierarchically. At the time of this writing, research on HME recognition has been ongoing for approximately 50 years. In this paper we focused upon techniques published in the last 10 years. Substantial improvements in recognition accuracy were seen over the past decade: first, through improvements in structural approaches (e.g., using stochastic context-free grammars), followed by additional improvements with the introduction of Deep Neural Networks (DNNs) , particularly encoder–decoder models generating text representations of formula structure (e.g., in LaTeX), and Graph Neural Network (GNN)-based approaches. Early work in the decade focused on online data in pen- and touch-based input devices, while more recently strong techniques for recognizing handwritten formulas from images have emerged (i.e., offline recognition). The CROHME competitions and their associated benchmarking datasets have provided a common research platform. Commercial applications now incorporate HME recognition as well, which we feel is a positive trend.

Presently, HME models using encoder–decoder models are the most accurate. In addition to the underlying models, this is also attributable to the use of ensemble models, language models, hybrid models, and data augmentation and generation methods used in training. We expect the accuracy of these systems to improve steadily as more training data becomes available.

However, several opportunities for improving HME recognition systems remain. At the intersection of pattern recognition and linguistics, HME recognition researchers have had difficulty designing language models (LMs) that represent HMEs effectively for recognition. One-dimensional context processing, including basic n-gram models and recent RNN-LMs, have been used to increase recognition rates. Because HMEs have two-dimensional structure, efficiently evaluating two-dimensional LMs that capture context between symbols in two dimensions may produce larger improvements than obtained using one-dimensional LMs, and so this seems like a promising direction for future work. Other opportunities include improving HME recognition through public datasets of data from specific user groups (e.g., school children), user customization/adaptation, and improved user interfaces for math formula entry and editing.

ACKNOWLEDGMENT

This work is partially supported by the Grant-in-Aid for Scientific Research JP24H00738, JP22H00085, JP21K18136, and JP21K17761.

REFERENCES

[1] R. H. Anderson, "Syntax-directed recognition of hand-printed two-dimensional mathematics," in *Proc. Association for Computing Machinery Inc. Symposium*, Washington, USA, 1967.

[2] S. K. Chang, "A method for the structural analysis of two-dimensional mathematical expressions," *Information Science: an International Journal*, pp. 253-272, 1970.

[3] A. Belaid, J.-P. Haton, "A syntactic approach for handwritten mathematical formula recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 1, pp. 105-111, 1984.

[4] H. Mouchère, C. Viard-Gaudin, D. H. Kim, J. H. Kim, G. Utpal, "CROHME2011: Competition on recognition of online handwritten mathematical expressions," in *Proc. 11th International Conference on Document Analysis and Recognition*, Beijing, China, 2011.

[5] H. Mouchère, R. Zanibbi, U. Garain, C. Viard-Gaudin, "Advancing the state of the art for handwritten math recognition: the CROHME competitions, 2011-2014," *International Journal on Document Analysis and Recognition*, vol. 19, no. 2, pp. 173-189, 2016.

[6] R. Zanibbi, D. Blostein, "Recognition and retrieval of mathematical expressions," *International Journal of Document Analysis and Recognition*, vol. 15, no. 4, pp. 331-357, 2012.

[7] D. Blostein, A. Grbavec, "Recognition of mathematical notation," in *Handbook of character recognition and document image analysis*, World Scientific, 1997, pp. 557-582.

[8] K. Chan, D. Yeung, "Mathematical expression recognition: a survey," *International Journal of Document Analysis and Recognition*, vol. 3, no. 1, pp. 3-15, 2000.

[9] D. Zhelezniakov, V. Zaytsev, O. Radyvonenko, "Online handwritten mathematical expression recognition and applications: A survey," *IEEE Access*, vol. 9, pp. 38352-38373, 2021.

[10] J. Zhang, J. Du, S. Zhang, D. Liu, Y. Hu, J. Hu, S. Wei, L. Dai, "Watch, attend and parse: an end-to-end neural network based approach to handwritten mathematical expression recognition," *Pattern Recognition*, vol. 71, pp. 196-206, 2017.

[11] J. Zhang, J. Du, L. Dai, "Track, Attend and Parse (TAP): An End-to-end Framework for Online Handwritten Mathematical Expression Recognition," *IEEE Transactions on Multimedia*, vol. 21, no. 1, pp. 221-233, 2019.

[12] J.-W. Wu, F. Yin, Y.-M. Zhang, X.-Y. Zhang, C.-L. Liu, "Image-to-Markup Generation via Paired Adversarial Learning," in *Proc. Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Dublin, Ireland, 2018.

[13] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE transactions on acoustics, speech, and signal processing*, vol. 3, pp. 328-339, 1989.

[14] F. Álvaro, J. Sánchez, J. Benedí, "Recognition of on-line handwritten mathematical expressions using 2D stochastic context-free grammars and hidden Markov models," *Pattern Recognition Letters*, vol. 35, pp. 58-67, 2014.

[15] T. Zhang, H. Mouchère, C. Viard-Gaudin, "A tree-BLSTM-based recognition system for online handwritten mathematical expressions," *Neural Comput. Appl.*, vol. 32, pp. 196-206, 2020.

[16] T. Zhang, H. Mouchere, C. Viard-Gaudin, "Online handwritten mathematical expressions recognition by merging multiple 1D interpretations.," in *Proc. 15th International Conference on Frontiers in Handwriting Recognition.*, 2016.

[17] MyScript, "MyScript - Handwriting technology & digital ink solutions," [Online]. Available: https://www.myscript.com. [Accessed 10 08 2023].

[18] Wiris, "Wiris - Digital solutions for Math and Science Education," [Online]. Available: https://www.wiris.com. [Accessed 10 8 2023].

[19] D. Zhelezniakov, V. Zaytsev, O. Radyvonenko, "Acceleration of Online Recognition of 2D Sequences Using Deep Bidirectional LSTM and Dynamic Programming," in *Proc. International Work-Conference on Artificial Neural Networks*, 2019.

[20] H. Mouchère, C. Viard-Gaudin, R. Zanibbi, U. Garain, "ICFHR 2014 competition on recognition of on-line handwritten mathematical expressions (CROHME 2014)," in *Proc. 14th International Conference on Frontiers in Handwriting Recognition*, Heraklion, Greece, 2014.

[21] A. D. Le, M. Nakagawa, "A system for recognizing online handwritten mathematical expressions by using improved structural analysis," *International Journal on Document Analysis and Recognition*, vol. 19, no. 4, pp. 305-319, 2016.

[22] R. Yamamoto, S. Sako, T. Nishimoto, S. Sagayama, "Online recognition of handwritten mathematical expressions based on stroke-based stochastic context-free grammar," in *Proc. 7th International Conference on Document Analysis and Recognition*, Edinburgh, UK, 2006.

[23] F. Alvaro, R. Zanibbi, "A shape-based layout descriptor for classifying spatial relationships in handwritten math," in *Proc. 2013 ACM symposium on Document engineering*, 2013.

[24] F. Julca-Aguilar, N. S. Hirata, H. Mouchère, C. Viard-Gaudin, "Subexpression and dominant symbol histograms for spatial relation classification in mathematical expressions," in *Proc. 23rd International Conference on Pattern Recognition*, 2016.

[25] K.-F. Chan, D.-Y. Yeung, "Error detection, error correction and performance evaluation in on-line mathematical expression recognition," *Pattern Recognition Letters*, vol. 34, no. 8, pp. 1671-1684, 2001.

[26] D. Prusa, V. Hlavac, "Mathematical formulae recognition using 2D grammars," in *Proc. 9th International Conference on Document Analysis and Recognition*, Curitiba, Brazil, 2007.

[27] R. Zanibbi, D. Blostein, J. R. Cordy, "Recognizing mathematical expressions using tree transformation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 11, pp. 1455-1467, 2002.

[28] Y. Eto, M. Suzuki, "Mathematical formula recognition using virtual link network," in *Proc. 6th International Conference on Document Analysis and Recognition*, Seattle, USA, 2001.

[29] W. Chmielnicki, K. Stapor, "Investigation of normalization techniques and their impact on a recognition rate in handwritten numeral recognition," *Scheda Informaticae*, vol. 19, pp. 53-77, 2010.

[30] C. L. Liu, K. Nakashima, H. Sako, H. Fujisawa, "Handwritten digit recognition: investigation of normalization and feature extraction techniques," *Pattern Recognition*, vol. 37, no. 2, pp. 265-279, 2004.

[31] S. MacLean, G. Labahn, "A new approach for recognizing handwritten mathematics using relational grammars and fuzzy sets," *International Journal on Document Analysis and Recognition*, vol. 16, no. 2, pp. 1-25, 2013.

[32] A. M. Awal, H. Mouchre, C. Viard-Gaudin, "A global learning approach for an online handwritten mathematical expression recognition system," *Pattern Recognition Letters*, vol. 35, pp. 68-77, 2014.

[33] P. A. Chou, "Recognition of equations using a two-dimensional stochastic context-free grammar," in *Proc. SPIE 1199, Visual Communications and Image Processing IV*, Philadelphia, USA, 1989.

[34] F. Simistira, V. Katsouros, G. Carayannis, "Recognition of online handwritten mathematical formulas using probabilistic SVMs and stochastic context-free grammars," *Pattern Recognition Letters*, vol. 53, no. C, pp. 85-92, 2015.

[35] M. Celik, B. Yanikoglu, "Probabilistic mathematical formula recognition using a 2D context-free graph grammar," in *Proc. 11th International Conference on Document Analysis and Recognition*, Beijing, China, 2011.

[36] A. Kosmala, G. Rigoll, S. Lavirotte, L. Pottier, "On-line handwritten formula recognition using hidden Markov models and context-dependent graph grammars," in *Proc. 5th International Conference on Document Analysis and Recognition*, Bangalore, India, 1999.

[37] F. D. Julca-Aguilar, H. Mouchere, C. Viard-Gaudin, N. S. T. Hirata, "Top-down online handwritten mathematical expression parsing with graph grammar," in *Proc. Iberoamerican Congress on Pattern Recognition*, Montevideo, Uruguay, 2015.

[38] Y. Shi, H. Li, F. Soong, "A unified framework for symbol segmentation and recognition of handwritten mathematical expressions," in *Proc. 9th International Conference on Document Analysis and Recognition*, Curitiba, Brazil, 2007.

[39] M. Koschinski, H.-J. Winkler, M. Lang, "Segmentation and recognition of symbols within handwritten mathematical expressions," in *Proc. 1995 International Conference on Acoustics, Speech, and Signal Processing*, Detroit, USA, 1995.

[40] H.-J.Winkler, M. Lang, "Online symbol segmentation and recognition in handwritten mathematical expressions," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, Munich, Germany, 1997.

[41] H.-J. Winkler, M. Lang, "Symbol segmentation and recognition for understanding handwritten mathematical expressions," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, Munich, Germany, 1997.

[42] L. Hu, R. Zanibbi, "MST-based visual parsing of online handwritten mathematical expressions," in *Proc. 15th International Conference on Frontiers in Handwriting Recognition*, Shenzhen, China, 2016.

[43] A. Shah, R. Zanibbi, "Line-of-Sight with Graph Attention Parser (LGAP) for Math Formulas," in *Proc. 17th International Conference on Document Analysis and Recognition*, San Jose, USA, 2023.

[44] T.-N. Truong, H. T. Nguyen, C. T. Nguyen, M. Nakagawa, "Learning Symbol Relation Tree for Online Handwritten Mathematical Expression Recognition," in *Proc. Asian Conference on Pattern Recognition*, 2022.

[45] S. MacLean, G. Labahn, "A Bayesian model for recognizing handwritten mathematical expressions," *Pattern Recognition*, vol. 48, no. 8, pp. 2433-2445, 2015.

[46] T. H. Rhee, J. H. Kim, "Efficient search strategy in structural analysis for handwritten mathematical expression recognition," *Pattern Recognition Letters,* vol. 42, no. 12, pp. 3192-3201, 2009.

[47] K. M. Phan, A. D. Le, B. Indurkhya, M Nakagawa, "Augmented incremental recognition of online handwritten mathematical expressions," *International Journal on Document Analysis and Recognition,* vol. 21, no. 4, pp. 253-268, 2018.

[48] K. M. Phan, C. T. Nguyen, A. D. Le, M. Nakagawa, "An incremental recognition method for online handwritten mathematical expressions," in *Proc. 3rd IAPR Asian Conference on Pattern Recognition*, Kuala Lumpur, Malaysia, 2015.

[49] B. Radakovic, G. Predovic and B. Dresevic, "Geometric Parsing of Mathematical Expressions". USA Patent US8064696 B2, December 2011.

[50] G. Predovic, A. Abdulkader, B. Dresevic, P. A. Viola and M. Vukosavljevic, "Recognition of mathematical expressions". USA Patent US8009915 B2, 30 August 2011.

[51] B. Q. Vuong, S. C. Hui, Y. He, "Progressive structural analysis for dynamic recognition of on-line handwritten mathematical expressions," *Pattern Recognition Letters,* vol. 29, no. 5, pp. 647-655, 2008.

[52] E. Shelhamer, J. Long, T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. 39, p. 640–651, 2017.

[53] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems,* vol. 30, 2017.

[54] O. Vinyals, Ł. Kaiser, T. Koo, S. Petrov, I. Sutskever, G. Hinton, "Grammar as a foreign language," in *Proc. Advances in Neural Information Processing Systems 2015*, Montreal, Canada, 2015.

[55] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *Proc. 2016 IEEE International Conference on Acoustics, Speech and Signal Processing*, Shanghai, China, 2016.

[56] M.-T. Luong, I. Sutskever, Q.V. Le, O. Vinyals, W. Zaremba, "Addressing the Rare Word Problem in Neural Machine Translation," in *Proc. 7th International Joint Conference on Natural Language Processing*, Beijing, China, 2015.

[57] T. Truong, C.T. Nguyen, K.M. Phan, M. Nakagawa, "Improvement of End-to-End Offline Handwritten Mathematical Expression Recognition by Weakly Supervised Learning," in *Proc. 17th Int. Conf. Front. Handwrit. Recognit.*, Dortmund, Germany, 2020.

[58] J.W. Wu, F. Yin, Y.M. Zhang, X.Y. Zhang, C.L. Liu, "Handwritten Mathematical Expression Recognition via Paired Adversarial Learning," *Int. J. Comput. Vis.*, pp. 1-16, 2020.

[59] Y. Deng, A. Kanervisto, J. Ling, A. M. Rush, "Image-to-Markup Generation with Coarse-to-Fine Attention," in *Proc. 34th International Conference on Machine Learning*, Sydney, Australia, 2017.

[60] A. D. Le, M. Nakagawa, "Training an end-to-end system for handwritten mathematical expression recognition by generated patterns," in *Proc. 14th IAPR International Conference on Document Analysis and Recognition*, Kyoto, Japan, 2017.

[61] J. W. Wu, F. Yin, Y.-M. Zhang, X. Y. Zhang, C. L. Liu, "Graph-to-Graph: Towards Accurate and Interpretable Online Handwritten Mathematical Expression Recognition," in *Proc. AAAI Conference on Artificial Intelligence*, 2021.

[62] J. Tang, H. Guo, J. Wu, F. Yin, L. Huang, "Offline handwritten mathematical expression recognition with graph encoder and transformer decoder," *Pattern Recognition,* vol. 148, p. 110155, 2024.

[63] H. D. Nguyen, A. D. Le, M. Nakagawa, "Recognition of online handwritten math symbols using deep neural networks," *IEICE TRANSACTIONS on Information and Systems,* vol. 99, no. 12, pp. 3110-3118, 2016.

[64] F. Álvaro, J. -A. Sánchez, J. -M. Benedí, "Offline Features for Classifying Handwritten Math Symbols with Recurrent Neural Networks,," in *Proc. 22nd International Conference on Pattern Recognition*, Stockholm, Sweden, 2014.

[65] C. T. Nguyen, T. N. Truong, H. T. Nguyen, M. Nakagawa, "Global Context for Improving Recognition of Online Handwritten Mathematical Expressions," in *Proc. 16th International Conference on Document Analysis and Recognition, ICDAR*, 2021.

[66] M. Mahdavi, R. Zanibbi, "Visual Parsing with Query-Driven Global Graph Attention (QD-GGA): Preliminary Results for Handwritten Math Formula Recognition," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. 2020.*, 2020.

[67] C. T. Nguyen, H. T. Nguyen, K. Morizumi, M. Nakagawa, "Temporal classification constraint for improving handwritten mathematical expression recognition," in *Proc. 15th International Conference on Document Analysis and Recognition*, Lausanne, Switzerland, 2021.

[68] B. Li, Y. Yuan, D. Liang, X. Liu, Z. Ji, J. Bai, W. Liu, X. Bai, "When Counting Meets HMER: Counting-Aware Network for Handwritten Mathematical Expression Recognition," 2022.

[69] Z. Li, X. Wang, Y. Liu, L. Jin, Y. Huang, K. Ding, "Improving Handwritten Mathematical Expression Recognition Via Similar Symbol Distinguishing," *IEEE Transactions on Multimedia*, pp. 1-13, 2023.

[70] B. Hambly, T. Lyons, "Uniqueness for the signature of a path of bounded variation and the reduced path group," *Annals of Mathematics*, vol. 171, pp. 109--167, 2010.

[71] H. Y. Guo, C. Wang, F. Yin, H. Y. Liu, J. W. Wu, C. L. Liu, "Primitive Contrastive Learning for Handwritten Mathematical Expression Recognition," in *Proc. 26th International Conference on Pattern Recognition*, 2022.

[72] A.D. Le, B. Indurkhya, M. Nakagawa, "Pattern generation strategies for improving recognition of Handwritten Mathematical Expressions," *Pattern Recognit. Lett.*, vol. 128, pp. 255-262, 2019.

[73] Y. Yuan, X. Liu, W. Dikubab, H. Liu, Z.Li, Z. Wu, X.Bai, "Syntax-Aware Network for Handwritten Mathematical Expression Recognition," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[74] Z. Liu, Y. Yuan, Z. Ji, J. Bai, X. Bai, "Semantic Graph Representation Learning for Handwritten Mathematical Expression Recognition," in *Proc. 17th International Conference on Document Analysis and Recognition*, San Jose, USA, 2023.

[75] A. D. Le, "Recognizing Handwritten Mathematical Expressions via Paired Dual Loss Attention Network," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020.

[76] J. Wang, J. Du, J. Zhang, B. Wang, B. Ren, "Multi-modal attention network for handwritten mathematical expression recognition," in *Proc. International Conference on Document Analysis and Recognition*, 2019.

[77] J. Wang, J. Du, J. Zhang, B. Wang, B. Ren, "Stroke constrained attention network for online handwritten mathematical expression recognition," *Pattern Recognition*, vol. 119, 2021.

[78] M. Mahdavi, R. Zanibbi, H. Mouchère, "ICDAR 2019 CROHME + TFD : Competition on Recognition of Handwritten Mathematical Expressions and Typeset Formula Detection," in *Proc. 15th Int. Conf. Doc. Anal. Recognit.*, 2019.

[79] Y. Xie, H. Mouchère, F. Simistira Liwwicki, S. Rakesh, R. Saini, M. Nakagawa, C. T. Nguyen, T. N. Truong, "ICDAR 2023 CROHME: Competition on Recognition of Handwritten Mathematical Expressions," in *17th International Conference on Document Analysis and Recognition*, San Jose, USA, 2023.

[80] H. Ding, K. Chen, Q. Huo, "An encoder-decoder approach to handwritten mathematical expression recognition with multi-head attention and stacked decoder," in *Proc. 16th International Conference on Document Analysis and Recognition*, Lausanne, Switzerland, 2021.

[81] W. Zhao, L. Gao, Z. Yan, S. Peng, L. Du, Z. Zhang, "Handwritten Mathematical Expression Recognition with Bidirectionally Trained Transformer," in *Proc. 16th International Conference on Document Analysis and Recognition*, Lausanne, Switzerland, 2021.

[82] X. Bian, B. Qin, X. Xin, J. Li, X. Su, Y. Wang, "Handwritten mathematical expression recognition via attention aggregation based bi-directional mutual learning," in *Proc. AAAI Conference on Artificial Intelligence*, 2022.

[83] W. Zhao, L. Gao, "CoMER: Modeling Coverage for Transformer-Based Handwritten Mathematical Expression Recognition," in *Proc. European Conference on Computer Vision*, 2022.

[84] Q. Lin, X. Huang, N. Bi, C.Y. Suen, J. Tan, "CCLSL: Combination of Contrastive Learning and Supervised Learning for Handwritten Mathematical Expression Recognition," in *Proc. Asian Conference on Computer Vision*, 2022.

[85] C. Wang, H. Luo, X. Rao, W. Hu, N. Bi, J. Tan, "Relative Position Embedding Asymmetric Siamese Network for Offline Handwritten Mathematical Expression recognition," in *Proc. 17th International Conference on Document Analysis and Recognition*, San Jose, USA, 2023.

[86] T.N. Truong, H.Q. Ung, H.T. Nguyen, C.T. Nguyen, M. Nakagawa, "Relation-based representation for handwritten mathematical expression recognition," in *Proc. International Conference on Document Analysis and Recognition*, Korea, 2021.

[87] T. N. Truong, C. T. Nguyen, M. Nakagawa, "Syntactic data generation for handwritten mathematical expression recognition," *Pattern Recognition Letters*, vol. 153, pp. 83-91, 2022.

[88] H. Q. Ung, C. T. Nguyen, H. T. Nguyen, M. Nakagawa, "A Transformer-Based Math Language Model for Handwritten Math Expression Recognition," in *Proc. 16th International Conference on Document Analysis and Recognition, ICDAR-DIL*, 2021.

[89] J. Zhang, J. Du, Y. Yang, Y.S. Si, W. lirong, "A Tree-Structured Decoder for Image-to-Markup Generation," in *Proc. 37th International Conference on Machine Learning*, 2020.

[90] J. Zhang, J. Du, Y. Yang, Y.-Z. Song, L. Dai, "SRD: A Tree Structure Based Decoder for Online Handwritten Mathematical Expression Recognition," *IEEE Trans. Multimed.*, vol. 1, pp. 1-10, 2020.

[91] J. Wang, Q. Wang, J. Du, J. Zhang, B. Wang, B. Ren, "MRD: A Memory Relation Decoder for Online Handwritten Mathematical Expression Recognition," in *Proc. 16th International Conference on Document Analysis and Recognition*, Lausanne, Switzerland, 2021.

[92] C. Wu, J. Du, Y. Li, J. Zhang, C. Yang, B. Ren, Y. Hu, "TDv2: A Novel Tree-Structured Decoder for Offline Mathematical Expression Recognition," in *Proc. 36th AAAI Conference on Artificial Intelligence*, 2022.

[93] T.G. Dietterich, "Ensemble methods in machine learning, in:," in *Proc. International Work- shop on Multiple Classifier Systems - Proceedings 1. Springer Berlin Heidelberg*, Cagliari, Italy, 2000.

[94] B. Zoph, E.D. Cubuk, G. Ghiasi, T.-Y. Lin, J. Shlens, Q. V. Le, "Learning Data Augmentation Strategies for Object Detection," 2019.

[95] B. Chen, B. Zhu, M. Nakagawa, "Training of an on-line handwritten Japanese character recognizer by artificial patterns," *Pattern Recognit. Lett.*, vol. 35, pp. 178-185, 2014.

[96] C. Wigington, S. Stewart, B. Davis, B. Barrett, B. Price, S. Cohen, "Data Augmentation for Recognition of Handwritten Words and Lines Using a CNN-LSTM Network," in *Proc. Int. Conf. Doc. Anal. Recognit.*, Kyoto, Japan, 2017.

[97] Z. Li, L. Jin, S. Lai, Y. Zhu, "Improving Attention-Based Handwritten Mathematical Expression Recognition with Scale Augmentation and Drop Attention," in *Proc. Int. Conf. Front. Handwrit. Recognit.*, Dortmund, Germany, 2020.

[98] A. Graves, "Generating Sequences With Recurrent Neural Networks," in *http://arxiv.org/abs/1308.0850*, 2013.

[99] E. Alonso, B. Moysset, R. Messina, "Adversarial generation of handwritten text images conditioned on sequences," in *Proc. Int. Conf. Doc. Anal. Recognit.*, Sydney, Australia, 2019.

[100] S. MacLean, G. Labahn, E. Lank, M. Marzouk, D. Tausky, "Grammar-based techniques for creating groundtruthed sketch corpora," *International Journal Document Analysis and Recognition*, vol. 12, no. 1, pp. 65-74, 2011.

[101] V. T. M. Khuong, U. Q. Huy, N. Masaki, M.K. Phan, "Generating synthetic handwritten mathematical expressions from a LaTeX sequence or a mathML script," in *Proc. Int. Conf. Doc. Anal. Recognit.*, Sydney, Australia, 2019.

[102] C. Yang, J. Du, J. Zhang, C. Wu, M. Chen, J. Wu, "Tree-based data augmentation and mutual learning for offline handwritten mathematical expression recognition," *Pattern Recognition*, vol. 132, 2022.

[103] H. Mouchère, C. Viard-Gaudin, D. H. Kim, J. H. Kim, G. Utpal, "ICFHR 2012 competition on recognition of on-line mathematical expressions (CROHME 2012)," in *Proc. 13rd International Conference on Frontiers in Handwriting Recognition*, Bari, Italy, 2012.

[104] H. Mouchère, C. Viard-Gaudin, R. Zanibbi, U. Garain, D. H. Kim, J. H. Kim, "ICDAR 2013 CROHME: third international competition on recognition of online handwritten mathematical expressions," in *Proc. 12th International Conference on Document Analysis and Recognition*, Washington, USA, 2013.

[105] H. Mouchère, C. Viard-Gaudin, R. Zanibbi, U. Garain, "ICFHR2016 CROHME: competition on recognition of online handwritten mathematical expressions," in *Proc. 15th International Conference on Frontiers in Handwriting Recognition*, Shenzhen, China, 2016.

[106] K. Sain, A. Dasgupta, U. Garain, "EMERS: a tree matching-based performance evaluation of mathematical expression recognition systems," *International Journal on Document Analysis and Recognition*, vol. 14, no. 1, pp. 75-85, 2011.

[107] F. Álvaro, J.-A. Sánchez, J.-M. Benedí, "Unbiased evaluation of handwritten mathematical expression recognition," in *Proc. 13rd International Conference on Frontiers in Handwriting Recognition*, Bari, Italy, 2012.

[108] R. Zanibbi, A. Pillay, H. Mouchere, C. Viard-Gaudin, D. Blostein, "Stroke-based performance metrics for handwritten mathematical expressions," in *Proc. 11th International Conference on Document Analysis and Recognition*, Beijing, China, 2011.

[109] T. Zhang, H. Mouchère, C. Viard-Gaudin, "Tree-based BLSTM for mathematical expression recognition," in *Proc. 14th International Conference on Document Analysis and Recognition*, Kyoto, Japan, 2017.

[110] C. Gao, Y. Liu, S. Yao, J. Bai, X. Bai, L. Jin, C.-L. Liu, "ICDAR 2023 Competition on Recognition of Multi-line Handwritten Mathematical Expressions," San Jose, CA, 2023.

[111] W. Ohyama, M. Suzuki, S. Uchida, "Detecting mathematical expressions in scientific document images using a u-net trained on a diverse dataset," *IEEE Access*, vol. 7, pp. 144030-144042, 2019.

[112] H. Oka, H. T. Nguyen, C. T. Nguyen, M. Nakagawa, T. Ishioka, "Fully Automated Short Answer Scoring of the Trial Tests for Common Entrance Examinations for Japanese University," in *Proc. 24th International Conference on Artifical Intelligence in Education*, Durham, UK, 2022.

[113] H. T. Nguyen, C. T. Nguyen, H. Oka, T. Ishioka, M. Nakagawa, "Handwriting recognition and automatic scoring for descriptive answers in Japanese language tests," in *Proc. 18th International Conference on Frontiers in Handwriting Recognition*, Hyderabad, India, 2022.

[114] D. Giordano, A. M. Florea, "Proceedings of the First International Workshop on Pen-Based Learning Technologies," in *International Workshop on Pen-Based Learning Technologies*, Catania, Italy, 2007.

[115] M. Nakagawa, N. Lozano, H. Oda, "Paper architecture and an exam scoring application," in *Proc. 1st International Workshop on Pen-Based Learning Technologies*, 2007.

[116] K. Koyama, M. Nakagawa, "Implementation of a pen and paper based exam marking system," in *Proc. World Conference on E-Learning in Corporate, Government, Healthcare & Higher Education*, Orland, USA, 2010.

[117] N. T. Heffernan, C. L. Heffernan, "The ASSISTments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching," *International Journal of Artificial Intelligence in Education*, vol. 24, pp. 470-497, 2014.

[118] MathNet, "MathNet," [Online]. Available: https://www.etrialstestbed.org/projects/mathnet-competition. [Accessed 1 12 2023].

[119] V. T. M. Khuong, H. Q. Ung, C. T. Nguyen, M. Nakagawa, "Clustering Offline Handwritten Mathematical Answers for Computer-Assisted Marking," in *Proc. Inter. Conf. Pattern Recognit. Artif. Intell.,*, Montreal, 2018.

[120] H. Q. Ung, C. T. Nguyen, K. M. Phan, V. T. M. Khuong, M. Nakagawa, "Clustering online handwritten mathematical expressions," *Pattern Recognition Letters*, vol. 146, pp. 267-275, 2021.

[121] C. T. Nguyen, V. T. M. Khuong, H. T. Nguyen, M. Nakagawa, "CNN based spatial classification features for clustering offline handwritten mathematical expressions," *Pattern Recognition Letters*, vol. 131, pp. 113-120, 2020.

[122] S. Medjkoune, H. Mouchère, S. Petitrenaud, & C. Viard-Gaudin, "Combining Speech and Handwriting Modalities for Mathematical Expression Recognition," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 2, pp. 259-272, 2017.

[123] J. Zhang, J. Du, L. Dai, "Multi-Scale Attention with Dense Encoder for Handwritten Mathematical Expression Recognition.," in *Proc. 24th international conference on pattern recognition.*, 2018.

[124] D.-H. Wang; F. Yin; J.-W. Wu; Y.-P. Yan; Z.-C. Huang; G.-Y. Chen; Y. Wang; C.-L. Liu, "ICFHR 2020 Competition on Offline Recognition and Spotting of Handwritten Mathematical Expressions - OffRaSHME," in *Proc. 17th International Conference on Frontiers in Handwriting Recognition*, Dortmund, Germany, 2020.

[125] F. D. J. Aguilar, N. S. Hirata, "Expressmatch: a system for creating ground-truthed datasets of online mathematical expressions," in *Proc. 10th IAPR International Workshop on Document Analysis Systems*, Queensland, Australia, 2012.

[126] S. Quiniou, H. Mouchère, S. Saldarriaga, C. Viard-Gaudin, E, Morin, S. Petitrenaud, S. Medjkoune, "Hamex - a handwritten and audio dataset of mathematical expressions," in *Proc. 11th International Conference on Document Analysis and Recognition*, Beijing, China, 2011.

[127] G. Labahn, E. Lank, M. Marzouk, A. Bunt, S. MacLean, D. Tausky, "MathBrush: a case study for pen-based interactive mathematics," in *Proc. 5th Eurographics conference on Sketch-Based Interfaces and Modeling*, Annecy, France, 2008.

[128] J. Stria, M. Bresler, D. Prusa, V. Hlavc, "Mfrdb: Database of annotated on-line mathematical formulae," in *Proc. 13th International Conference on Frontiers in Handwriting Recognition*, Bari, Italy, 2012.

[129] A.-M. Awal, H. Mouchère, C. Viard-Gaudin, "Towards handwritten mathematical expression recognition," in *Proc. International Conference on Document Analysis and Recognition*, Barcelona, Spain, 2009.

835