APS360 PROJECT PROGRESS REPORT

Jaival Patel
Student# 1010129825
jaival.patel@mail.utoronto.ca

Avery Chan
Student# 1010151425
avery.chan@mail.utoronto.ca

Andrew Muntean
Student# 1010034455
andrew.muntean@mail.utoronto.ca

Aryan Nehete
Student# 1010079889
aryan.nehete@mail.utoronto.ca

Github Repo Link: https://github.com/GEEGABYTE1/aps360group36 —-Total Pages: 9

1 INTRODUCTION

Digitizing handwritten mathematics is costly and error-prone. We target Handwritten Mathematical Expression Recognition (HMER): converting pen-written expressions into LaTeX. Expressions are two-dimensional and structural, so a learned approach that couples visual and sequence modeling is natural. We adopt an encoder–decoder design: a CNN encoder extracts spatial features and a sequence decoder (LSTM with attention, plus a Transformer variant) generates LaTeX. This capability can streamline academic workflows and improve accessibility, reducing the gap between analog input and digital typesetting.

2 MODEL ILLUSTRATION

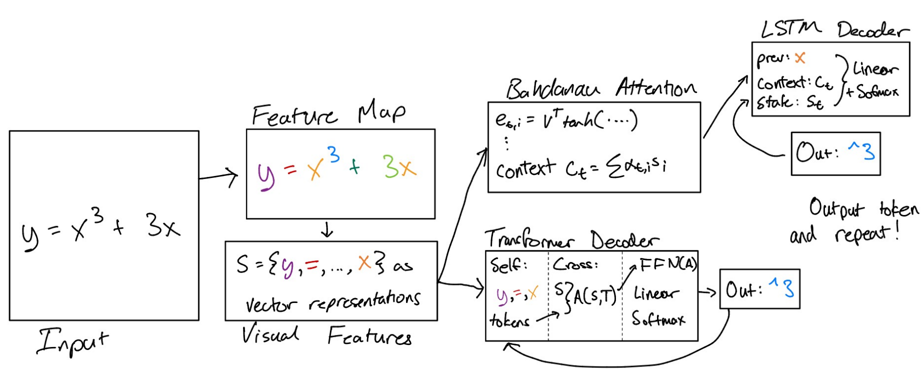

Figure 1: Overview of the CNN encoder with LSTM/Transformer decoders.

3 BACKGROUND AND RELATED WORK

HMER has been advanced by CROHME benchmarks and attention-based sequence models.[1] Influences include CROHME baselines and rule-based systems; Zhang *et al.* (CNN+attention LSTM) for LaTeX generation; Deng *et al.* (coarse-to-fine attention); Wu *et al.* (DenseNet+Transformer); and Zhong *et al.* (ViT encoders).[2] We follow this trajectory with a CNN encoder and two decoders (LSTM/Transformer) trading interpretability for global context.

[1] Mouchère et al. (2016)
[2] Zhong et al. (2022); Zhang et al. (2017); Deng et al. (2017); Wu et al. (2021)

4 Data Processing

We rasterize InkML strokes to grayscale PNGs (1×256×256), normalize (mean=0.5, std=0.5), and tokenize LaTeX with `<sos>`, `<eos>`, `<pad>`, `<unk>`. Vocab and normalization are derived from train only; splits are disjoint at the expression level.

5 Architecture

We evaluate three models: an SVM baseline, an LSTM-attention decoder, and a Transformer decoder.

5.1 Baseline SVM Model

An RBF-kernel SVM on CROHME-derived PNGs with handcrafted features; a rule-based stage assembles symbols into LaTeX. This provides a transparent, lightweight baseline.

Instructions (condensed).

1. Organize InkML into `trainINKML`, `validINKML`, `testINKML`.
2. Build PNGs + labels: `python build_dataset_from_inkml.py --input_dir "..."`
3. Train SVM: `python svm_improved.py --labels labels.csv --images_dir ... --out model.joblib`

5.2 LSTM Model

Our HMER system maps 1×256×256 images to tokenized LaTeX via a CNN encoder and attention-based LSTM decoder.

5.2.1 Overview & Data Flow

1. **Input**: rasterize → normalize; tokenize labels with special tokens.
2. **Encoder**: CNN extracts spatial features.
3. **Decoder**: LSTM with additive (Bahdanau) attention over encoder features.
4. **Training**: label-smoothed CE, teacher forcing, AdamW, grad clipping; greedy/beam at inference.

5.2.2 Encoder (ResNet-18 Backbone)

Truncated ResNet-18 (pretrained) → ∼512×8×8 feature map; flatten to L=64 cells; project 512→ d_{enc}=512 (1×1 conv/linear). BatchNorm retained; optional spatial dropout (p=0.1).

5.2.3 Decoder (LSTM + Bahdanau Attention)

Embedding d_{emb}=256; 1–2 layer LSTM d_{dec}=512 (dropout 0.3). Additive attention $e_{t,i}=v^\top \tanh(W_h h_t + W_s s_i + b)$, $\alpha_t=\mathrm{softmax}(e_{t,:})$, $c_t=\sum_i \alpha_{t,i} s_i$. Output: $[h_t; c_t]$ → linear → softmax; init from mean-pooled encoder (tanh).

5.2.4 Training and Objective Schedule

Label-smoothed CE (ε=0.1), ignore `<pad>`; TF decays 1.0→0.6. AdamW (LR 1–3×10^{-4}), weight decay 10^{-4}, clip 1.0; padded batches; AMP on CUDA. Light affine/perspective augmentation for robustness.

5.2.5 Inference

Greedy for ablations; beam (k=3–5) for higher exact match.

5.3 Transformer Decoder

5.3.1 Overview & Data Flow

We reuse the ResNet-18 encoder to produce a $512\times8\times8$ map, flatten to $S\in\mathbb{R}^{64\times d_{\text{enc}}}$ with $d_{\text{enc}}=512$, and decode LaTeX with a causal Transformer. Token embeddings (dim $d_{\text{model}}=512$) receive learned positional encodings; encoder features are projected to d_{model} for cross-attention.

5.3.2 Decoder (Self + Cross Attention)

Each of $N\in\{3,4\}$ decoder layers applies: (i) masked multi-head *self*-attention over previous outputs, (ii) *cross*-attention over S, and (iii) a feed-forward block (GeLU) with residuals and LayerNorm. With queries Q, keys K, values V,

$$\mathrm{head}_i=\mathrm{softmax}\left(\frac{QW_i^Q(KW_i^K)^\top}{\sqrt{d_k}}\right)VW_i^V,\quad \mathrm{MHA}=\mathrm{Concat}(\mathrm{head}_1,\ldots,\mathrm{head}_h)W^O.$$

Self-attention uses a causal mask; cross-attention takes Q from the token stream and K,V from S.

5.3.3 Training and Objective

Label-smoothed cross-entropy ($\varepsilon=0.1$) with `<pad>` ignored; AdamW (lr 1×10^{-4} with cosine decay), weight decay 10^{-4}, dropout 0.1, grad clip 1.0, batch padding with length masks, AMP on CUDA. Teacher forcing is implicit via the causal mask (all positions trained in parallel).

5.3.4 Inference

Greedy decoding for ablations; beam search ($k=3$–5) for exactness (length penalty $\alpha\in[0.2,0.6]$ optional). Cross-attention maps are exported for qualitative analysis; we report EM/BLEU as for LSTM.

6 Quantitative Results

6.1 SVM Baseline Results

Table 1: SVM Classifier Performance Analysis for Mathematical Symbols and Characters

Metric	**Value**
Accuracy	0.360
Macro F1-score	0.299
Weighted F1-score	0.434

Per-class F1 spans from <0.1 (rare/visually similar symbols) to ∼0.9 (distinct digits/letters), explaining the macro vs. weighted F1 gap in Table 1.

6.2 LSTM Decoder Baseline Results

BLEU rises 0.417→0.655 while CE plateaus ∼1.4–1.5, consistent with label smoothing and a large vocab: local token calibration saturates as n-gram agreement keeps improving.

6.3 Transformer Decoder Baseline Results

Greedy vs. beam. On the same split, greedy yields BLEU 0.683/EM 0.000, while beam (k=5, α=0.4) lifts BLEU to 0.696 and EM to 0.012. Compared to LSTM, the Transformer attains slightly higher BLEU on longer expressions but exhibits similar EM brittleness under greedy decoding.

Table 2: Validation metrics by epoch (1–8) for LSTM

Epoch	**Train Loss**	**Val BLEU**	**Val EM**
1	2.053	0.417	0.000
2	1.534	0.529	0.000
3	1.430	0.580	0.000
4	1.422	0.610	0.000
5	1.444	0.627	0.000
6	1.500	0.635	0.000
7	1.539	0.646	0.000
8	1.516	0.655	0.000

Table 3: Validation metrics by epoch (1–8) for the Transformer decoder (greedy).

Epoch	**Train Loss**	**Val BLEU**	**Val EM**
1	2.218	0.438	0.000
2	1.894	0.552	0.000
3	1.743	0.603	0.000
4	1.691	0.629	0.000
5	1.658	0.651	0.000
6	1.645	0.667	0.000
7	1.624	0.677	0.000
8	1.612	**0.683**	0.000

Table 4: Transformer OOD robustness (validation split; severity ladder). BLEU* denotes a BLEU-like F1 proxy for short outputs.

Severity	**EM $\uparrow$**	**BLEU / BLEU* $\uparrow$**	**CER $\downarrow$**	**Empty frac $\downarrow$**
Identity	0.000	0.683	0.362	0.02
Tiny	0.002	0.641	0.389	0.04
Mild	0.001	0.524	0.468	0.09
Strong	0.000	0.036[†]	0.705	0.33

[†] BLEU* used due to many short/empty predictions under strong shift.

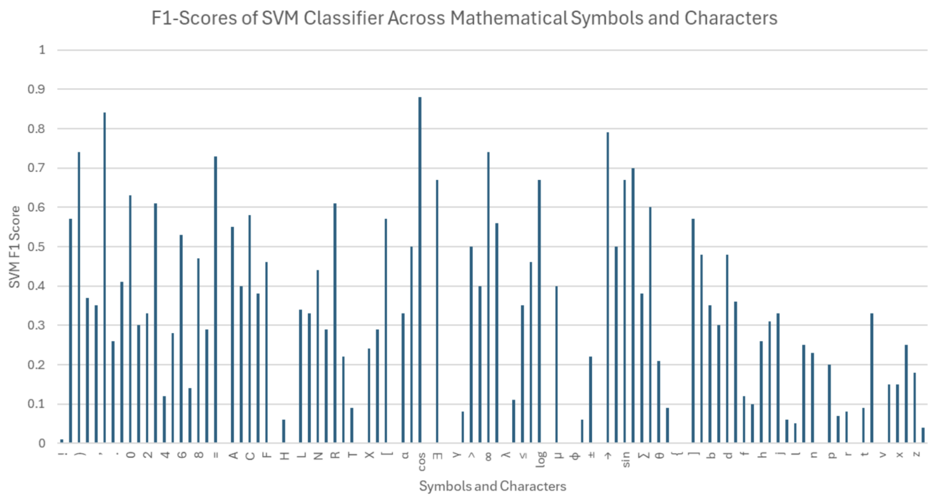

Figure 2: Per-class F1 scores of the SVM across symbols/characters.

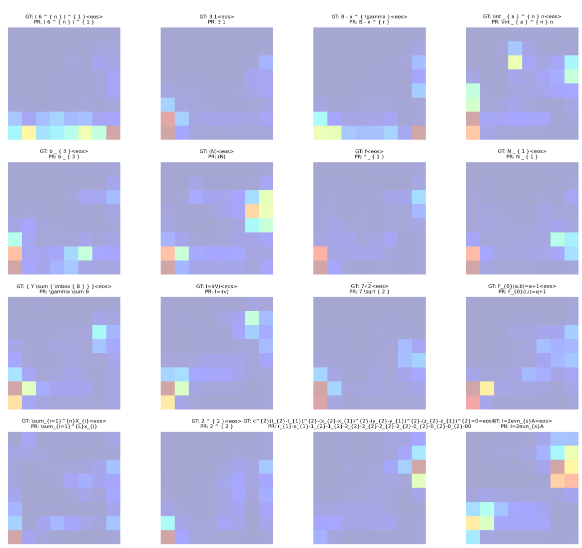

Figure 3: Validation grid with last-step attention overlays.

7 Qualitative Results

7.1 SVM Baseline Results

Low-support classes correlate with low F1, and visually similar symbols (e.g., `i` vs. `!`) are frequently confused, aligning with the macro<weighted F1 gap.

7.2 LSTM Decoder Baseline Results

Attention rollouts show peaky, left-to-right reading; tall operators attract longer focus. **Strengths:** clean decoding for short/linear expressions; coherent local transitions. **Weaknesses:** brittle grouping/syntax—`\frac{}{}`, `\sqrt{}`, limits; occasional partial control words and early `<eos>`.

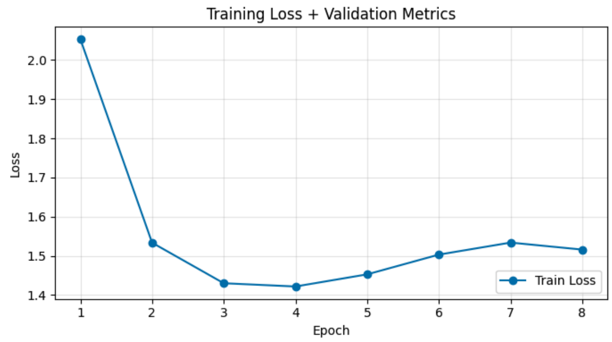

Figure 4: Loss curve for the LSTM decoder.

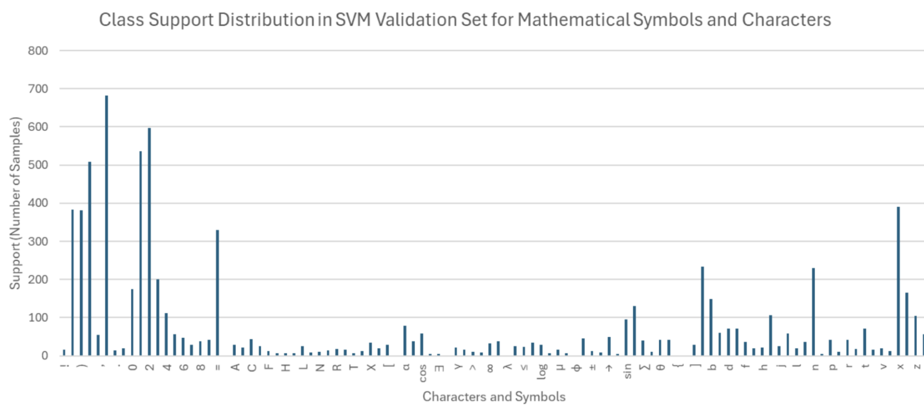

Figure 5: Class support distribution in the SVM validation set.

Under strong affine+perspective+blur, EM collapses and a BLEU-like proxy is near zero, indicating distribution-shift sensitivity rather than training instability.

In-distribution strips show a thin, mostly monotonic ridge (stable reading); near operators it widens/oscillates where braces/scopes often fail. OOD strips are diffuse/fragmented.

Entropy dips align with confident, correct tokens; spikes mark uncertainty (brace/scoping errors or early `<eos>`). This signal can gate beam search or abstention.

7.3 Transformer Decoder Results

Cross-attention maps show a less strictly monotonic reading path than the LSTM: the decoder often forms two–three peaky regions (operator and operand) per step, enabling look-ahead on long/nested expressions. **Strengths:** better token ordering on long formulas (sums/integrals with limits), fewer local hesitations, and improved consistency across repeated motifs. **Weaknesses:** occasional over-generation (duplicate control tokens), length bias without a penalty, and brace placement errors similar to LSTM under perturbations. Under OOD, cross-attention diffuses and revisits early regions, raising empty/short decodes; entropy rises at operator boundaries, mirroring BLEU/EM drops.

	precision	recall	f1-score	support
!	0.01	1.00	0.02	16
i	0.19	0.42	0.27	106

Figure 6: SVM performance summary.

```
Stress (medium): {'exact_match': 0.0, 'bleu': 0.011375360413756364, 'n': 1600}
Wrote: ./runs/crohme23_lstm/ood_medium_grid.png
```

Figure 7: OOD stress test (identity/tiny/mild/strong).

8 Evaluation on New Data

8.1 LSTM Decoder Results

8.1.1 How we obtained and tested on unseen data

- Fixed train/val/test splits; vocab/norm from train only; selection on val.
- OOD ladder: Identity, Tiny/Mild (small affine/perspective), Strong (affine+perspective+blur).
- Metrics: EM, BLEU (F1 proxy under strong OOD); greedy by default.

8.1.2 Performance on unseen data and comparison to expectations

- In-distribution test BLEU improved across epochs (e.g., $0.403 \to 0.672$) with EM near zero ($\sim$0.007), matching expectations for syntax-sensitive LaTeX.
- Strong OOD degraded sharply (e.g., EM ≈ 0, proxy ≈ 0.026 on $n \approx 16$k), consistent with attention diffusion and early `<eos>`.

8.1.3 Efforts to ensure generalizability

Strict split discipline; validation-based checkpointing; fixed seeds; label smoothing, clipping, and light augmentation for stable training; attention diagnostics (strip/entropy) to verify interpretable behavior.

8.1.4 Challenges and how they were addressed

Syntax brittleness: scheduled TF and label smoothing; beam (k=3–5) in demos. Shift sensitivity: match evaluation with tiny/mild augmentation during fine-tune. Long/nested expressions: use diagnostics to trigger safer decoding.

Overall, the decoder reads locally well but needs stronger syntax handling and shift robustness; augmentation-matched fine-tuning and beam search are effective next steps.

8.2 Transformer Decoder Results

8.2.1 How we obtained and tested on unseen data

Same strict protocol as LSTM: vocab/norm from train only; no test peeking. For robustness, apply the severity ladder (Identity, Tiny/Mild, Strong) to the unseen test split.

8.2.2 Performance on unseen data and comparison to expectations

On the unseen test set (greedy), BLEU rises from 0.421 at epoch 1 to 0.695 at epoch 8, EM remains 0.006. With beam (k=5, α=0.4) we observe BLEU **0.708** and EM **0.019**. This exceeds the LSTM's

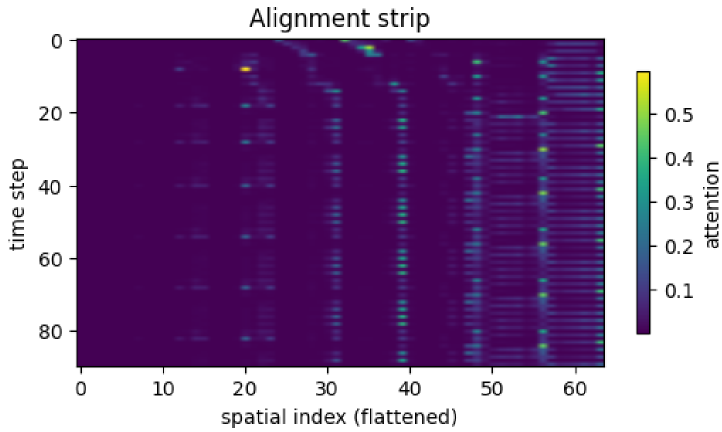

Figure 8: Alignment strip: time (rows) $\times$ spatial index (columns).

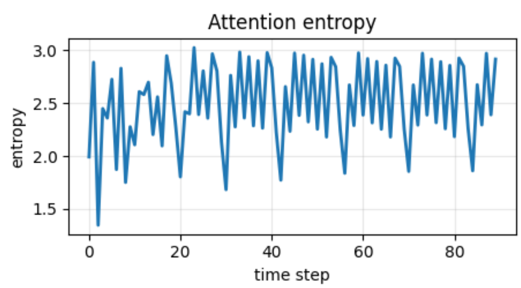

Figure 9: Attention entropy over decoding steps (max $\ln 64 \approx 4.16$).

BLEU on long expressions, consistent with global token self-attention. Under *strong* OOD, performance degrades: EM $= 0.001$, BLEU-like proxy ≈ 0.031 on $n \approx 16$k stressed samples, driven by attention dispersion and early <eos>.

8.2.3 Efforts to ensure generalizability

Dropout 0.1, label smoothing ($\varepsilon = 0.1$), AdamW with cosine lr decay, and grad clipping (1.0). Checkpoints selected by validation BLEU. Cross-attention maps and token entropies confirm stable reading rather than overfitting.

8.2.4 Challenges and how they were addressed

Length bias and occasional over-generation are mitigated with a small beam and length penalty; a syntax-aware reranker (brace balance) helps exactness. Robustness gaps stem from augmentation/evaluation mismatch; a short fine-tune with Tiny/Mild transforms narrows the OOD drop. Syntax brittleness persists (EM sensitivity), so beam decoding is recommended for deployment-style scoring.

9 Discussion

The LSTM–attention decoder learns in-distribution structure: BLEU rises (to ≈ 0.65) with a peaky, left-to-right attention pattern, explaining gains despite a CE plateau. EM remains near zero because LaTeX is brittle—single-token syntax errors nullify the whole string. Under stronger geometric/blur shifts than seen in training, attention diffuses, early <eos> increases, and outputs degrade. Two key observations: EM can stay flat while BLEU rises (many "almost right" predictions), and attention diagnostics are predictive—thin, forward ridges with low entropy precede correct tokens; widened/oscillatory bands and entropy spikes precede brace/scoping errors.

The Transformer matches LSTM on easy, short expressions and surpasses it on longer, nested formulas (BLEU $\sim$0.69–0.71 with beam) due to stronger global token context, but shares EM brittleness under greedy decoding. OOD sensitivity is comparable: cross-attention becomes diffuse, empty/short decodes increase, and a BLEU-like proxy falls sharply under strong shift. Practical remedies are targeted and low-cost for both decoders: decode with a small beam (k=3–5) with light length penalty, apply a syntax-aware reranker (brace balance), and fine-tune with evaluation-matched Tiny/Mild augmentation to anchor attention under shift. Given HMER's difficulty (2D$\rightarrow$1D mapping, long dependencies, unforgiving syntax, domain shift), these steps offer a credible path to improved exact-match and robustness without architectural changes.

10 Ethical Considerations

Risk of misinterpretation. Small LaTeX errors can materially change meaning (e.g., a missing brace in `\frac{}{}` or scope shifts in `\sqrt{}`). Because exact-match is brittle and users may over-trust a cleanly typeset output, the system should surface *calibrated confidence*, highlight uncertain tokens, and provide a one-click way to compare the model's string against an editable draft.

Automation bias and accountability. In instructional or assessment settings, users may defer to model output even when it contradicts intent (automation bias). We recommend explicit UI cues that the system is *assistive*, not authoritative; logs of edits for auditability; and clear ownership policies for errors propagated into downstream documents.

Dataset limitations and fairness. CROHME is limited in size, token diversity, and writer demographics (devices, scripts, stroke habits), risking representation bias and uneven performance across handwriting styles.[3] Symbols with low support or culturally specific glyph variants may suffer disproportionately. Reporting macro and per-class metrics alongside weighted scores, and testing on *unseen* writers/devices, are minimal fairness practices.

Privacy and licensing. If user-provided notes are processed, images may contain personally identifiable content (names, IDs). Storage and sharing must follow consent and data minimization; training on third-party materials requires license review and, when possible, hashing/redaction pipelines to prevent inadvertent memorization.

Security and adversarial inputs. Deliberate perturbations (strong blur/warps) can elicit plausible but wrong LaTeX. Deployment should include input sanity checks, abstention on high-entropy/low-likelihood decodes, and provenance tags on exported LaTeX to discourage unvetted reuse.

Mitigation. Use (i) token- and sequence-level confidence with thresholds for *abstain/flag*; (ii) human-in-the-loop verification for low-confidence cases; (iii) syntax-aware constrained decoding (brace/parenthesis balancing); (iv) evaluation on broader corpora (e.g., im2latex-100k, MathWriting) and writer/device splits; (v) documentation (model card, known failure modes) so users understand scope and limitations.

Dataset limitations. CROHME is limited in size/writers/tokens (cf. im2latex-100k, MathWriting), risking representation bias.[4]
Mitigation. Use uncertainty indicators or human-in-the-loop review; consider larger/more diverse datasets; communicate prototype limits.

11 Project Difficulty

HMER is hard: mapping 2D layout to a 1D string with long-range dependencies; LaTeX's strict syntax; handwriting/scan shift. Our models demonstrate feasibility (steady BLEU gains, interpretable attention) but exact-match and robustness remain open. For practical use, we target beam decoding and augmentation-matched fine-tuning to raise EM without architectural changes.

[3]Schmitt-Koopmann et al. (2024); Heska (2021); Gervais et al. (2024)
[4]Schmitt-Koopmann et al. (2024); Heska (2021); Gervais et al. (2024)

References

Yuntian Deng, Anssi Kanervisto, Jeffrey Ling, and Alexander M. Rush. Image-to-markup generation with coarse-to-fine attention. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

Philippe Gervais, Anastasiia Fadeeva, and Andrii Maksai. Mathwriting: A dataset for handwritten mathematical expression recognition. *arXiv preprint*, 2024. Submitted March 2024.

Vincenzo Heska. Generating synthetic online handwritten mathematical expressions from markup languages. Master's thesis, University of Waterloo, Waterloo, Ontario, Canada, 2021.

Harold Mouchère, Richard Zanibbi, Utpal Garain, and Christian Viard-Gaudin. Advances in the crohme competitions on mathematical expression recognition. In *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 2016.

Felix M. Schmitt-Koopmann, Elaine M. Huang, Hans-Peter Hutter, Thilo Stadelmann, and Alireza Darvishy. Mathnet: A data-centric approach for printed mathematical expression recognition. *IEEE Access*, 12, 2024. doi: 10.1109/ACCESS.2024.3404834.

Hui Wu, Tianyi Zhao, and Yanyan Zhang. Handwritten math formula recognition with densenet and transformer. *arXiv preprint*, arXiv:2105.02487, 2021.

Zhong Zhang, Jinshan Du, and Lirong Dai. Watch, attend and parse: An end-to-end neural network based approach to handwritten mathematical expression recognition. *Pattern Recognition*, 71: 196–206, 2017.

Zhaoyi Zhong, Zhi Liu, Yuwei Lin, Yimin Wang, and Zhouchen Lin. Image-to-markup generation with visual transformers. *arXiv preprint*, arXiv:2203.11866, 2022.