Character Recognition for Japanese Kanji

Matthew Geever, Salvador Palma, Thomas Buttigieg, Tiago Melo

10/06/2026 - Applied Machine Learning

UNIVERSITY OF COPENHAGEN

Problem Statement

- Japanese alphabet uses kanji: represent words or concepts
- Given a written sentence using kanji, can we recognize the characters and convert the sentence into digital text?
- Useful for digitization of books, documents.

(time)

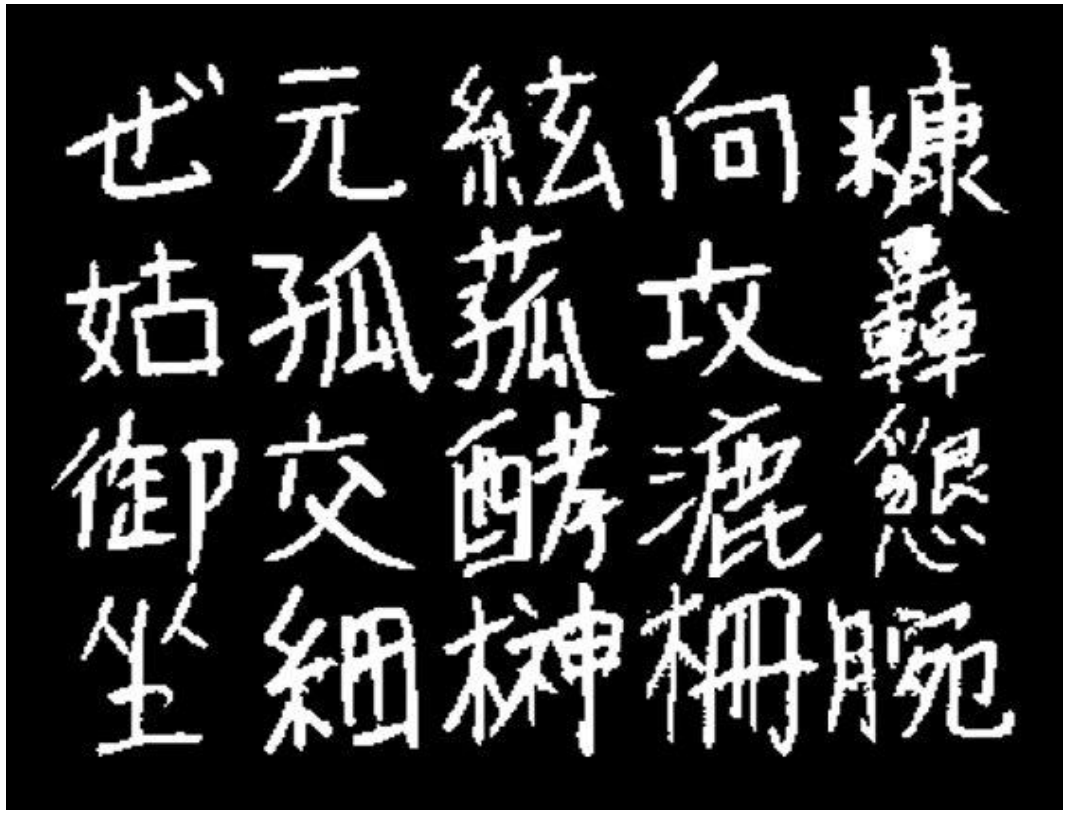
(high)

(country)

The Dataset: ETL9B

610K	3,036	~201	64×63
Training Samples	Kanji Classes	Samples per Class	Image Resolution (px)

- Japanese sentences can also have phonetic characters and punctuation: Need synthetic data for those (see Appendix)
- 3112 Total Classes



Recognizing Single Characters

- Convolutional Neural Network
- 3x Convolutional Layers (3x3 kernel)
- BatchNorm to stabilize training & reduce sensitivity to initial weights
- MaxPooling to downsample spatial dimensions
- Softmax to obtain probabilities

- Parameters: 18,242,344

Input: 64x63x1

Conv2D + BatchNorm + MaxPool: 32x31x64

Conv2D + BatchNorm + MaxPool: 16x15x128

Conv2D + BatchNorm + MaxPool: 8x7x256

Flatten Dense + Dropout (0.3): 1024

Softmax Output: 3112 Classes

Training Setup

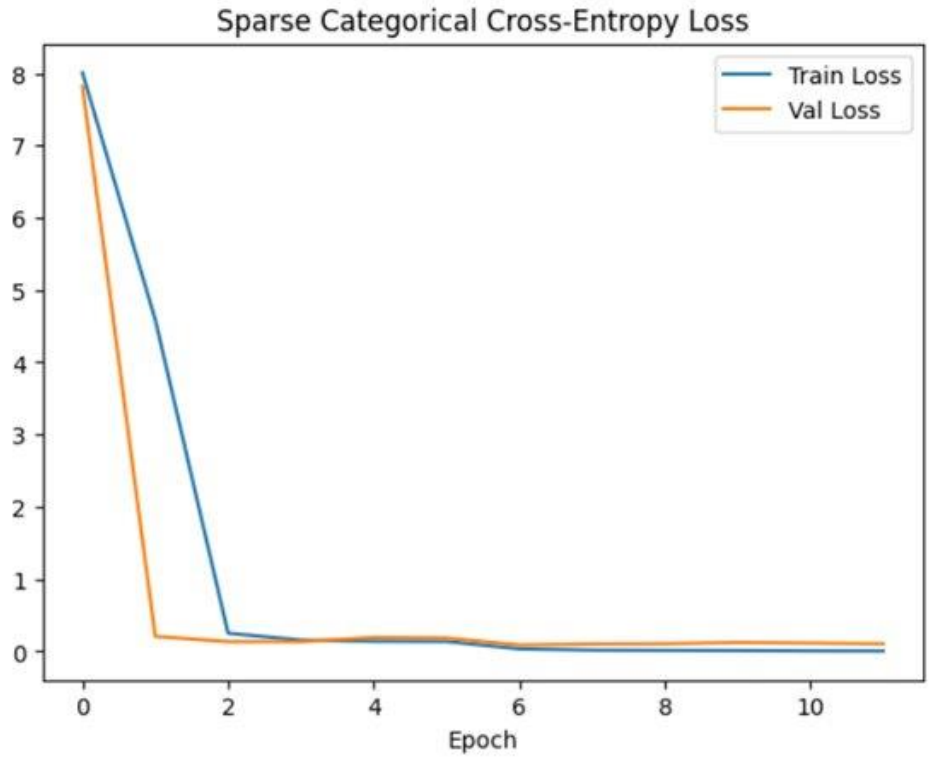
- Stratified 80/10/10 Training/Validation/Testing split
- Loss: Sparse Categorical Cross Entropy
- Optimizer: Adam
- Batch Size: 32
- Max Epochs: 50

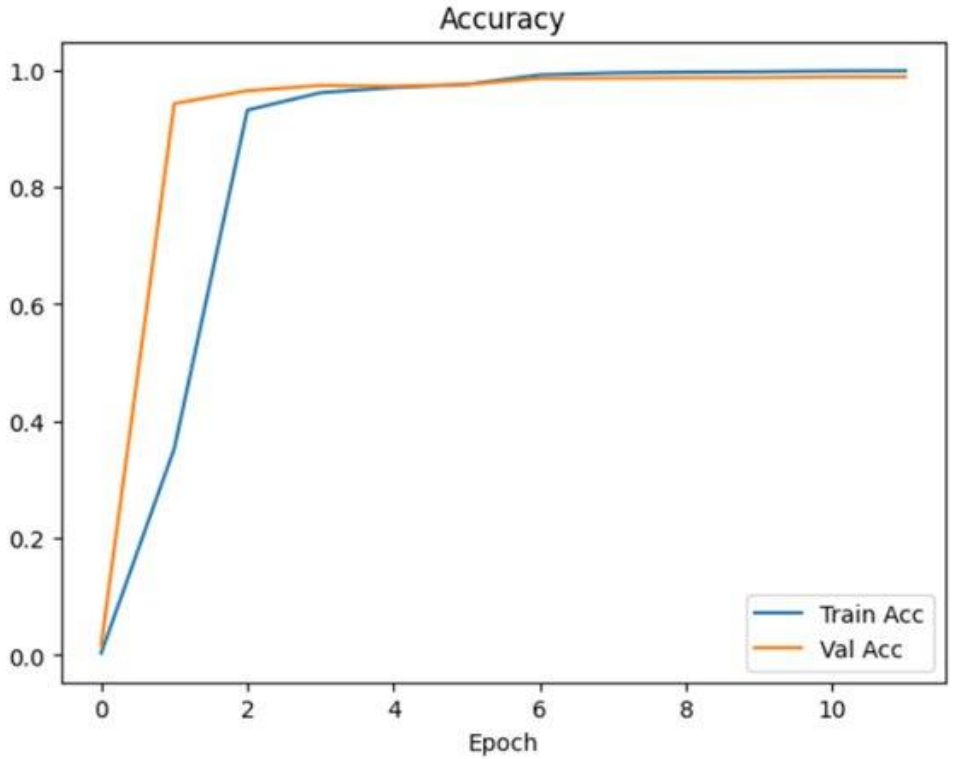
- Google Colab Pro – NVIDIA GPU (~50 minutes)

Training Performance + Classification Accuracy

~97%
Top-1 Test Accuracy

~99%
Top-5 Test Accuracy





Character Confusion Pairs

己 → 已 / 巳

Structural

Three enclosure variants with near-identical stroke count and structure — differ only in whether and where the enclosure is open

土 → 士

Resolution

Small length difference on the top horizontal stroke. At 64×63px resolution this distinction nearly vanishes

未 → 未

Structural

Top and bottom horizontal strokes switch relative length — the meaning (end vs not yet) flips with a single stroke proportion

大 → 犬

Single Stroke

A single dot added to 大 (big) creates 犬 (dog) — easily lost in noisy or quickly-written strokes

日 → 目

Resolution

One extra horizontal stroke inside the box separates sun from eye — often indistinguishable in compressed images

人 → 入

Resolution

Stroke angle difference only — diverging vs converging, nearly identical at training resolution and in handwriting

Processing Full Sentences

1 **Grayscale Load**

Read image as 8-bit grayscale

2 **Otsu Binarization**

Automatic global threshold — separates ink from paper without manual tuning of the threshold value

3 **Histogram Projection**

Sum ink pixels along rows (horizontal) or columns (vertical) depending on text orientation

4 **Line Detection**

Peaks in the row/col projection → line bounding bands; merge gap parameter fuses nearby active bands

5 **Character Detection**

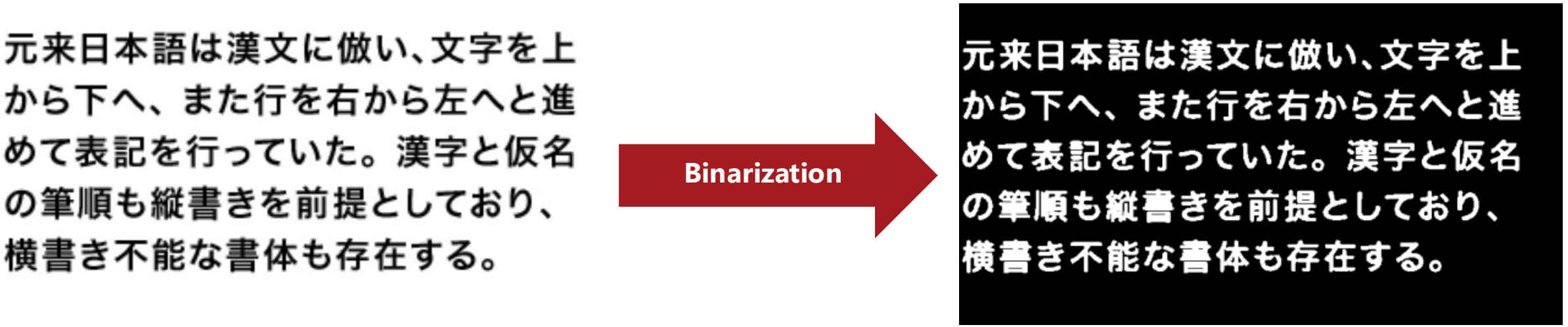
For each line strip, project again to find individual character bounding boxes

6 **Classify Segments**

Crop each bounding box, resize to 63×64px, pass through CNN → top-5 probabilities

Image Loading

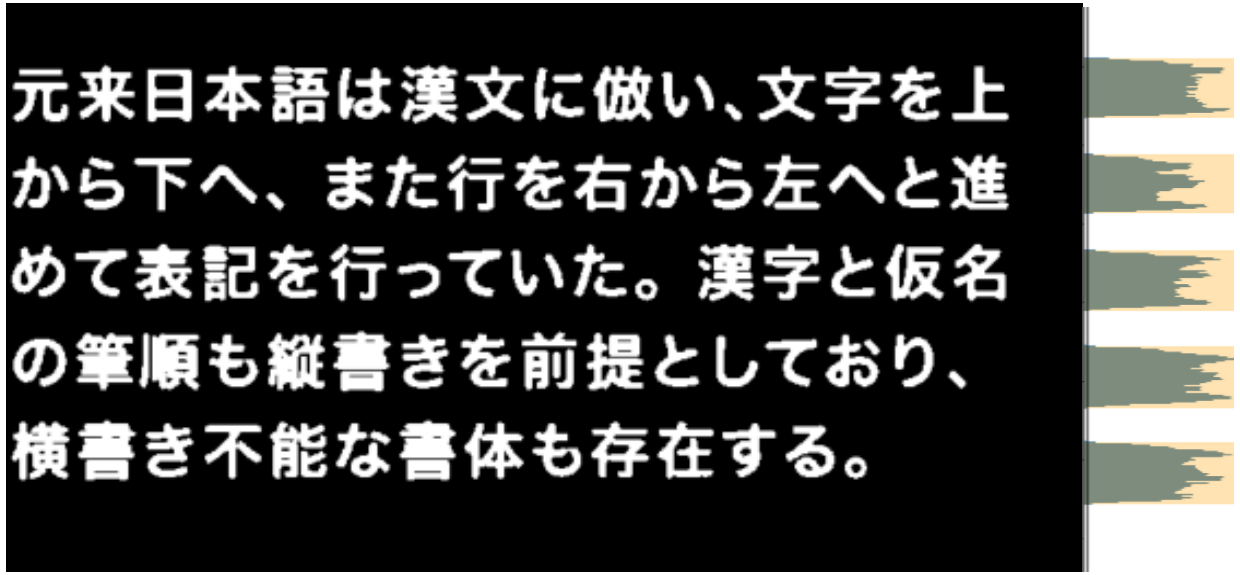
- Images are loaded as 8-bit grayscale and converted to inverted binary images



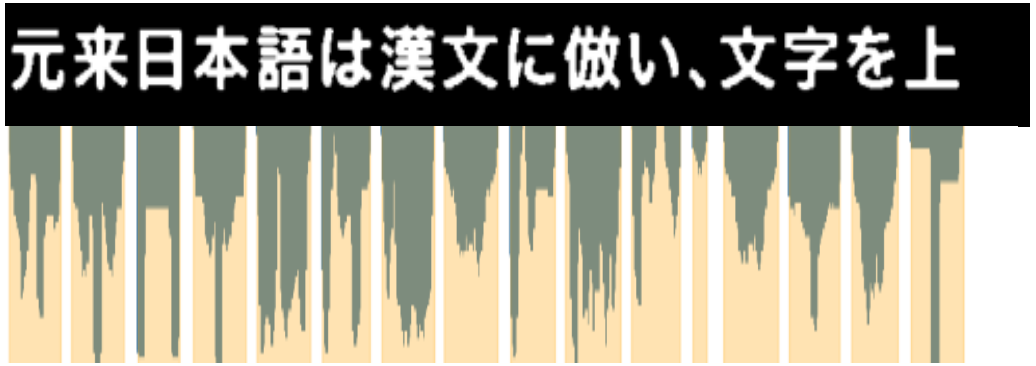
Segmentation

- Individual characters are identified using a 1D histogram projection along each axis

Line Segmentation

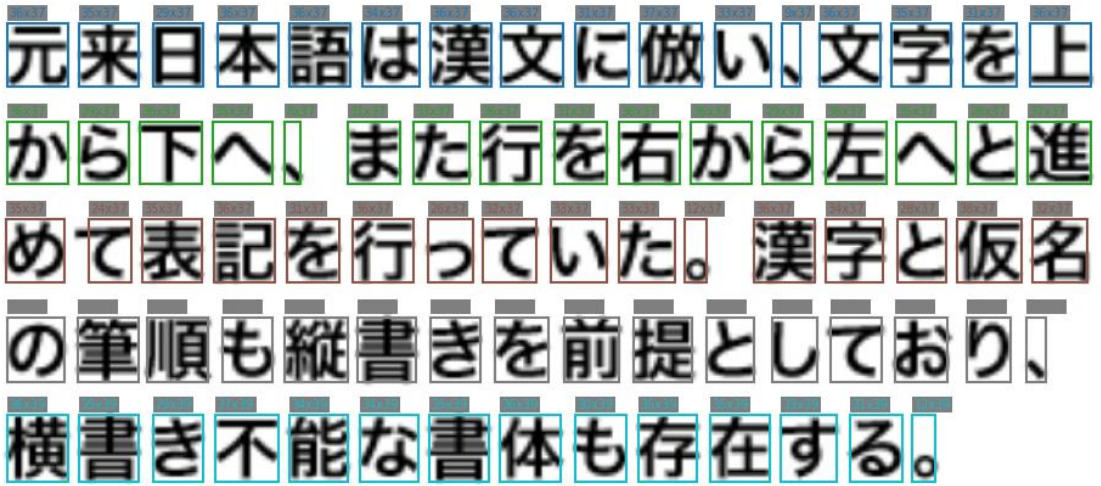


Character Segmentation



CNN Character Classification

Individual Character Segmentation



CNN Classification

元来日本語は漢攻に倣い、攻字を上から下へ、また行を右から左へと進めて張記を行つていた。漢字と仮名の筆順も縦書きを前提としてお口、横書き不能な書体も存在する。

- CNN still misclassifies some characters
- We need to add context for more robust character classification

BERT Comes To Help

- Context resolves ambiguity that pixel-level features cannot
- CL-Tohoku is a Character-level Japanese BERT pretrained on large Japanese corpora using Masked Language Modeling (MLM)
- Japanese has no spaces, so a character-level model avoids tokenization errors entirely
- Which kanji and kana naturally appear together in real Japanese text is exactly what is needed to disambiguate visually similar characters.

Reranking algorithm

1 **Assemble initial sequence**

Take CNN top-1 for each position → form a complete sentence guess

2 **For each position pos**

Substitute position pos with [MASK] token, keeping all other positions fixed

3 **Run BERT forward pass**

Feed masked sentence to BERT → softmax logits over full vocabulary at [MASK]

4 **Score each CNN candidate**

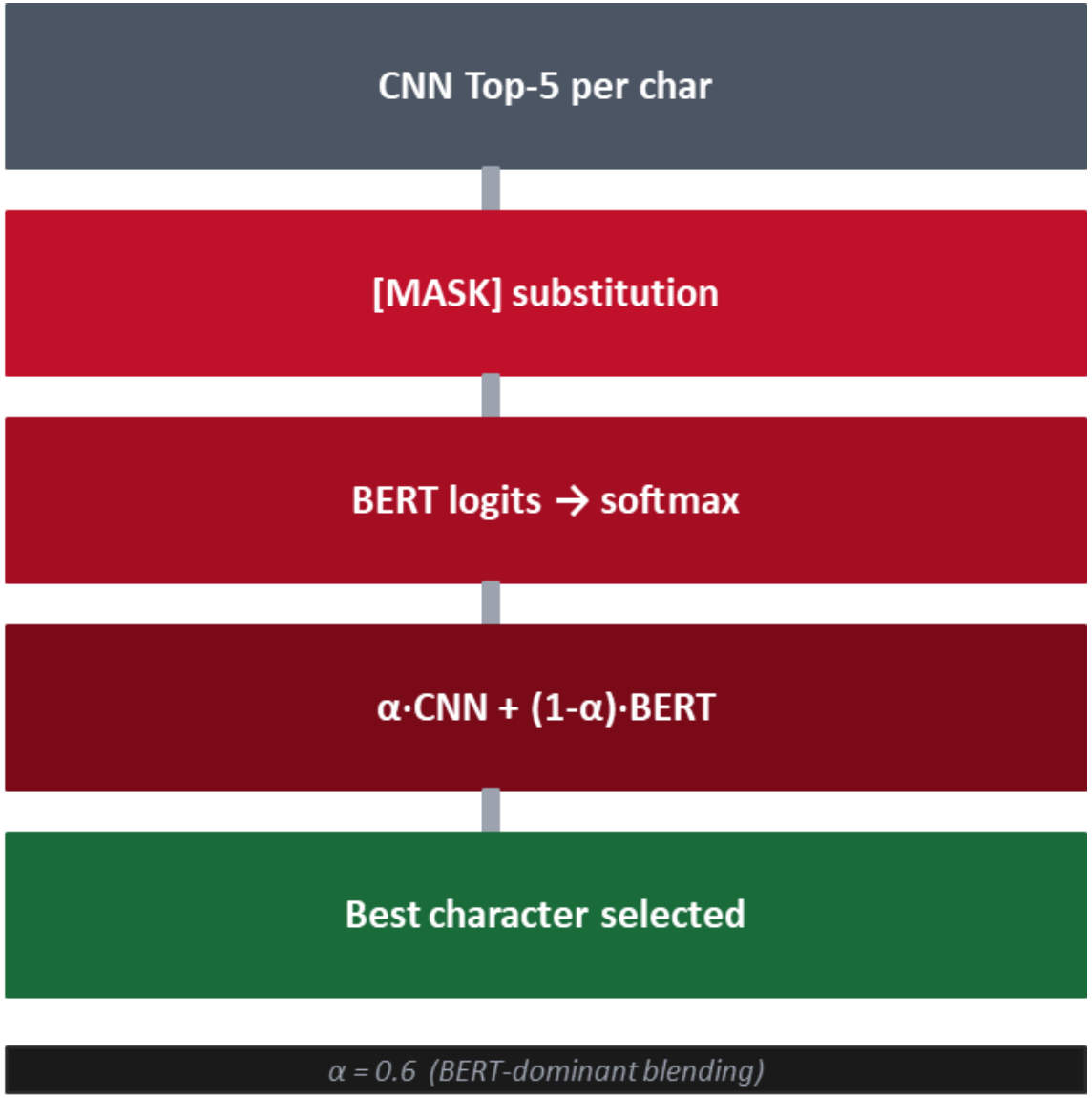
For each of the top-5 CNN candidates at pos, look up its BERT probability

5 **Fuse scores**

final_score = α × CNN_prob + (1-α) × BERT_prob with α = 0.6

6 **Select winner**

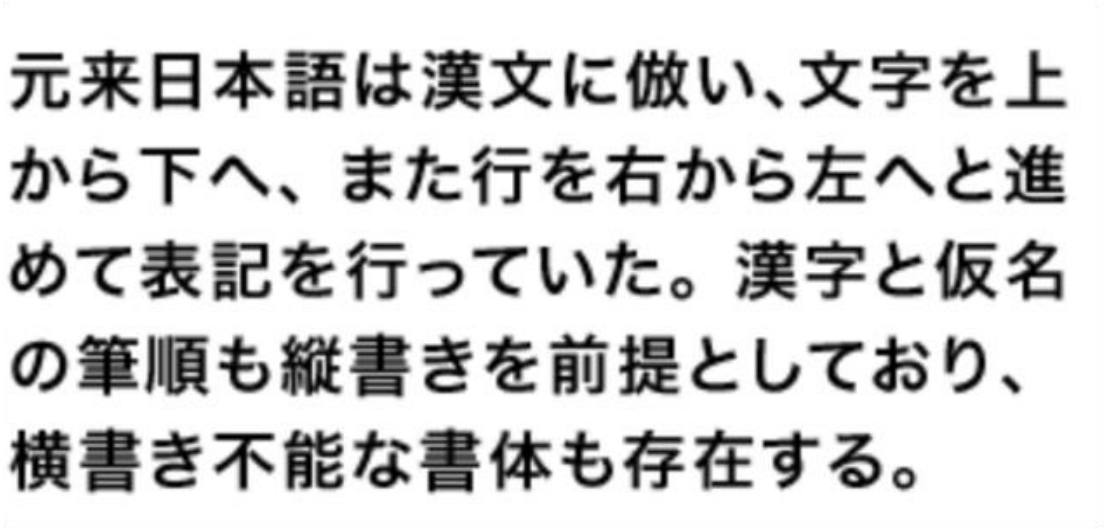
Character with highest fused score replaces CNN top-1 at this position



End-to-End Pipeline

↑ *Synthetic data extends CNN to 3,112 classes*

Results: Digital



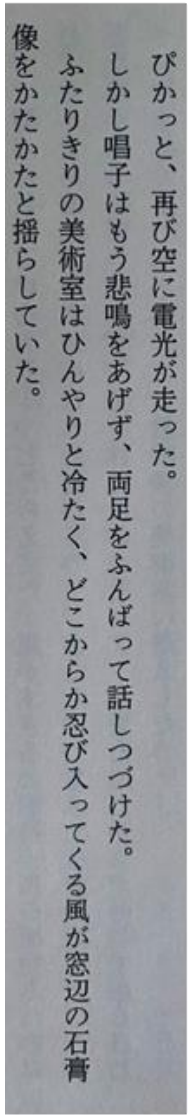
CNN output

元来日本語は漢攻に倣い、攻字を上
から下へ、また行を右から左へと進
めて張記を行つていた。漢字と仮名
の筆順も縦書きを前提としてお口、
横書き不能な書体も存在する。

BERT corrected

元来日本語は漢文に倣い、文字を上
から下へ、また行を右から左へと進
めて表記を行っていた。漢字と仮名
の筆順も縦書きを前提としておら、
横書き不能な書体も存在する。

Results: Printed Book


ぴかっと、再び空に電光が走った。
しかし唱子はもう悲鳴をあげず、両足をふんばって話しつづけた。
ふたりきりの美術室はひんやりと冷たく、どこからか忍び入ってくる風が窓辺の石膏像をかたかたと揺らしていた。


ぴかっと、再び空に電光が走った。
しかし唱子はもう悲鳴をあげず、両足をふんばって話しつづけた。
ふたりきりの美術室はひんやりと冷たく、どこからか忍び入ってくる風が窓辺の石膏像をかたかたと揺らしていた。

CNN output

ぴかつと、再び空に電光が走つた。

しかし唱乎はもう悲鴫をあげず、両足をふんばつて話しつづけた。

ふたりきりの瑳術室はひんやリと徐たく、どこからか忍び太つてくる風が窓辺の石鷲像をかたかたと揺らしていた。

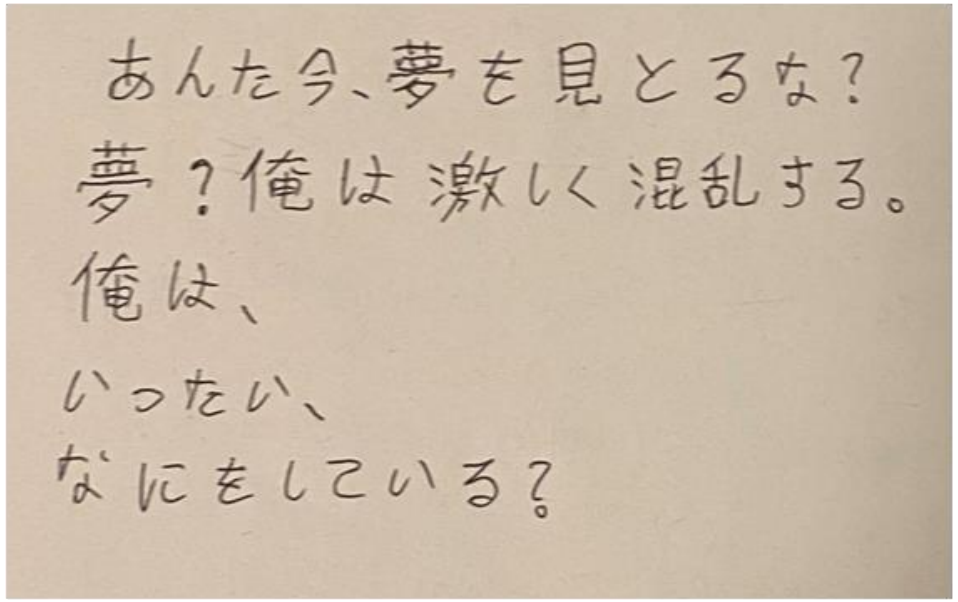
BERT corrected

ぴかっと、再び空に電光が走った。

しかし唱乎はもう悲鳴をあげず、両足をふんばって話しつづけた。

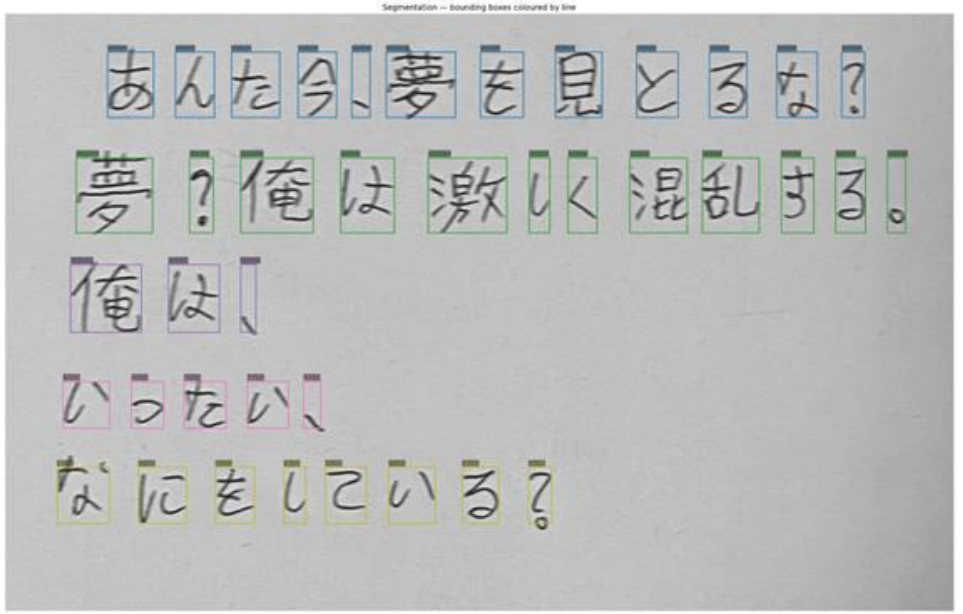
ふたりきりの瑳術室はひんやりと冷たく、どこからか忍び太ってくる風が窓辺の石鷲像をかたかたと揺らしていた。

Results: Handwritten



CNN output

あん左多、夢ぞ見とるな？
夢？俺は激しく湿乱まる。
俺は、
公つたい、
なにをしこいるで



BERT corrected

あんた今、夢を見とるな！
夢？俺は激しく混乱する。
俺は、
いったい、
なにをしている？

Results for Different Methods

Pipeline	Mean accuracy
Histogram + CNN	78.3%
Histogram + CNN + BERT	89.2%
OpenCV + CNN	55.0%
OpenCV + CNN + BERT	60.6%

We tried these different pipelines on text book text

Limitations

Segmentation parameter sensitivity

LINE_INK_THRESHOLD, LINE_MERGE_GAP, CHAR_MERGE_GAP, MAX_FATNESS and MAX_THINNESS all require manual tuning per image type (digital, printed, handwritten). A single parameter set rarely generalizes across all three — this is the biggest practical barrier to robust deployment.

ETL9B is handwriting-only → digital font gap

The CNN was trained exclusively on handwritten kanji. Clean digital fonts have different stroke weight distributions and glyph proportions. Some printed or digital characters fall outside the training distribution, causing reduced accuracy on those specific glyphs.

Font-rendered synthetic data ≠ handwritten katakana

Katakana and punctuation were added via font rendering with augmentation, but this doesn't capture the full natural variation of handwritten katakana. The model is noticeably weaker on hand-drawn katakana compared to kanji of similar complexity.

BERT struggles with short sentences

The reranking relies on surrounding context. A sentence of 3–5 characters provides very little context for BERT to work with — the masked position has almost no neighbors to condition on, reducing the benefit of reranking.

Author-specific or archaic writing styles

Literary Japanese often uses rare kanji readings, unconventional compounds, or archaic character forms not well-represented in either ETL9B or BERT's pretraining corpus. Both the CNN and BERT can fail on stylistically unusual text (e.g. classical literature, poetry, or authors with distinctive handwriting).

Future Directions

→ Contour/border-based character detection

Replace histogram projection with connected component analysis — find ink blobs directly and fit bounding boxes around them. Less sensitive to spacing assumptions and potentially more robust to overlapping strokes.

→ Expanded punctuation & symbol coverage

Currently covers 6 punctuation marks. Full Japanese text also uses 〜、…、（）、【】 and many others. Extending synthetic generation to cover all common symbols would improve sentence-level coverage.

→ Handwritten katakana training data

Collect or source real handwritten katakana samples (similar to ETL9B's methodology) to close the gap between font-rendered synthetic data and natural katakana handwriting variation.

→ Automatic parameter tuning for segmentation

Train a small classifier to predict optimal segmentation hyperparameters (threshold, merge gap) from image statistics — eliminating manual per-image tuning and enabling more robust batch processing.

→ Automatic rotation correction

Before segmentation, detect and correct page skew using Hough line transform or principal component analysis on ink pixels. This would make the histogram projection far more reliable without parameter re-tuning.

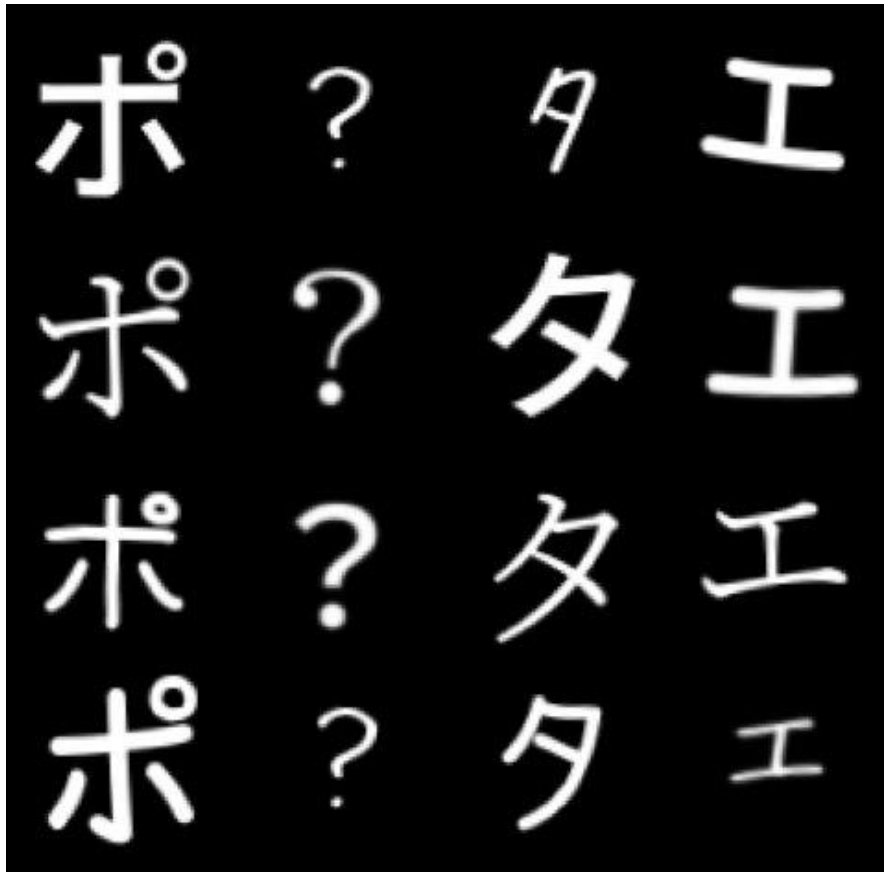
→ Distortion & perspective correction

Phone photos of books introduce keystone distortion. Applying homography correction (four-corner detection → warp) prior to Otsu binarization would handle real-world capture conditions.

APPENDIX

Synthetic Character Generation

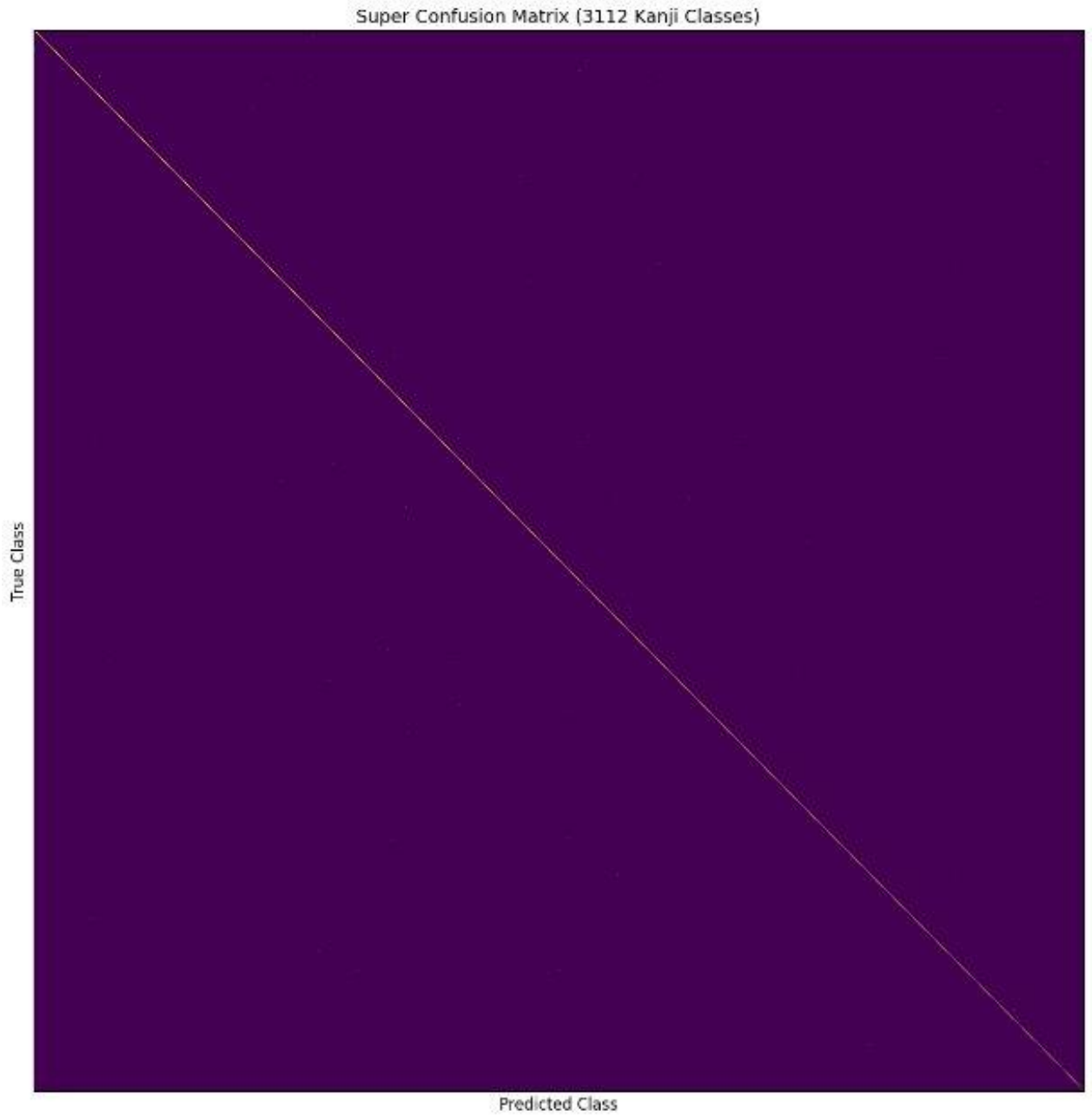
- ETL9B is a kanji-only dataset. A Japanese sentence also has phonetic characters (katakana) & punctuation: an added 76 characters.
- We generate using 6 different Japanese fonts.
- For each character:
  - Pick random font
  - Render using PIL
  - Add rotation + translation to simulate handwriting:
    - Rotation: between –5, +5 degrees
    - Translation: between –2, +2 pixels in each direction
  - Repeat 206 times
- Adds zero labeling cost



CNN Training

- Alongside the previously shown training setup, we also use:
  - LR Scheduler: ReduceLROnPlateau(patience=3, factor=0.2)
    This reduces the learning rate by a factor of 5 whenever we do not see any improvements for 3 consecutive epochs.
  - Early Stopping: monitor validation loss with patience=5, restoring best weights.
    Prevents overfitting without manual epoch tuning across 3112 classes.

Confusion Matrix



Top 10 Most Confused Kanji

之 | え | Count: 6

乎 | 平 | Count: 5

鳥 | 烏 | Count: 4

麿 | 磨 | Count: 4

酉 | 西 | Count: 4

採 | 探 | Count: 3

伸 | 仲 | Count: 3

味 | 昧 | Count: 3

釆 | 采 | Count: 3

膚 | 虜 | Count: 3

Segmentation Hyperparameters

	Sensitivity Impact
LINE_INK_THRESHOLD Fraction of max row ink required to mark a row as 'active'. Too low = noise lines. Too high = missed characters.	High
LINE_MERGE_GAP Max gap in pixels between active rows before splitting into two separate text lines. Critical for line spacing variation.	High
CHAR_INK_THRESHOLD Same as LINE threshold but applied column-wise within each line strip to detect individual character boundaries.	Med
CHAR_MERGE_GAP Max column gap before splitting into separate characters. Too small = one character split in two. Too large = two characters merged.	High
MAX_FATNESS Aspect ratio upper bound — rejects segments too wide to be a single character (merged strokes or noise blobs).	Low
MAX_THINNESS Aspect ratio lower bound — rejects segments too narrow to be a full character (partial strokes or punctuation remnants).	Low

Minimizable Characters

Certain katakana have "minimized" variants that are distinct characters but visually identical

つ → っ	よ → ょ	ゆ → ゅ	や → ゃ
ツ → ッ	ヨ → ョ	ユ → ュ	ヤ → ャ

When the CNN outputs a full-size character, but the bounding box is significantly smaller than the median character size for that line, the system checks the MINIMIZABLE_CHARACTERS map and considers the small variant as an additional candidate before passing to BERT.