Language-Grounded Continual Glyph Lexicon Retrieval for Ancient Chinese Character Recognition

Anonymous ACL submission

Abstract

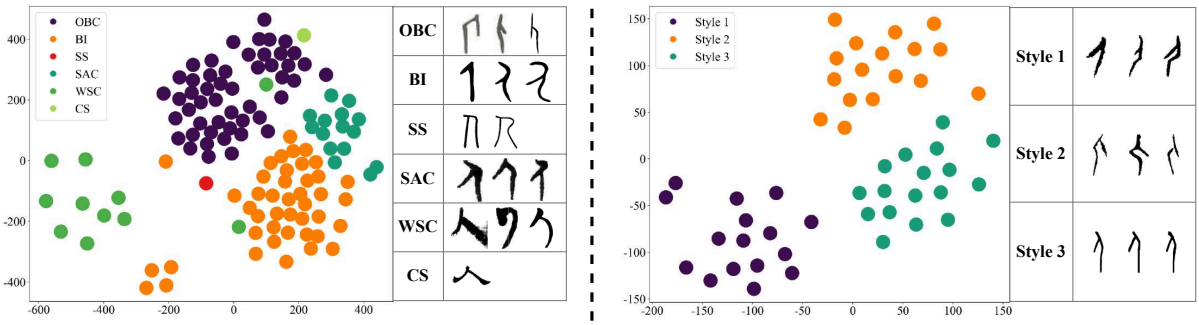
Ancient Chinese character recognition is commonly treated as closed-set visual classification, yet real cultural-heritage digitization workflows continually expand the searchable glyph lexicon and rely on textual knowledge such as dictionary meanings and paleographic descriptions. We formulate this setting as *language-grounded continual glyph lexicon retrieval*, where each class is a script-aware lexical entry grounded by glyph images and textual descriptions. We introduce **EvoCON**, a benchmark built on multi-script ancient Chinese glyph data with meaning descriptions, shape descriptions, a six-stage continual onboarding protocol, and a zero-shot candidate retrieval track. We also provide **AMR-CCR** as a strong reference model that combines frozen vision–language embeddings, script-conditioned lightweight adaptation, and a multi-prototype dictionary. Experiments show that EvoCON reveals challenges hidden by static closed-set evaluation, including continual lexicon growth, fine-grained inter-entry confusion, and strong intra-entry style variation. The results demonstrate that language-grounded dictionary retrieval is a practical paradigm for continually expanding ancient glyph lexicons.

1 Introduction

Ancient character recognition (Gordin et al., 2024; Fuentes-Ferrer et al., 2025; Swindall et al., 2021) is a key step toward making historical documents, stele rubbings, and excavated materials searchable and analyzable. In this paper, we focus on a highly representative branch—ancient Chinese Character Recognition (CCR)—where the task is distinguished by rich cross-script variations, long evolutionary trajectories, and a large character set.

Most existing studies on ancient CCR (Wang et al., 2026, 2025) adopt an offline closed-set classification setting, where a classifier is trained on a fixed set of categories and evaluated under the assumption that training and test data share the same label space. This formulation is convenient for static recognition, but it does not capture how cultural-heritage digitization is carried out in practice. Newly excavated and newly digitized materials are continuously incorporated into existing collections, introducing new scripts, new glyph forms, and new character identities over time (Mejías, 2025). Such workflows also frequently encounter rare or previously unseen characters, where experts must verify candidate identities by comparing glyph shapes, script styles, textual meanings, and historical evolution across scripts. As a result, the recognition target is not a fixed classification head, but an expanding glyph lexicon that must be updated, searched, and verified as new evidence becomes available.

We therefore study ancient CCR as *language-grounded continual glyph lexicon expansion*. Instead of treating each category as a classifier index, we treat it as a script-aware lexical entry grounded by glyph images, script identity, modern-character labels, dictionary meanings, and shape descriptions. Recognition is then performed as dictionary retrieval: a query glyph is matched against an expandable set of candidate entries in a shared embedding space. This paradigm is naturally compatible with continual onboarding, since new classes can be added as dictionary entries, and it also better matches scholarly workflows where ranked candidates with supporting visual or textual evidence are often more useful than a single closed-set prediction. Under this setting, a model must retrieve the correct entry from an expandable dictionary while handling two coupled challenges: (i) the number of entries grows over time, with subtle inter-entry differences and often scarce incremental samples; and (ii) the same entry may exhibit diverse writing styles across writers, carrier materials, and acquisition conditions, leading to strong intra-entry variation (Fig. 1).



(a) A single modern character across multiple scripts.

(b) One character within a single script under diverse writing styles.

Figure 1: Embedding distribution and glyph examples for a modern character.

To enable systematic evaluation, we introduce **EvoCON**, a language-grounded benchmark built upon EVOBC (Guan et al., 2024) with six historical scripts, script-aware labels, character-level meaning descriptions, image-level shape descriptions, a six-stage continual onboarding protocol, and a zero-shot candidate retrieval track. The zero-shot track matches image-only queries from held-out entries against textual candidate entries when no image prototypes are available, simulating cases where a newly observed glyph must be verified against a provided lexicon or hypothesized candidate set using linguistic and script-level evidence rather than unrestricted open-ended decipherment. We further provide **Anchored Modular Retrieval for Continual CCR (AMR-CCR)** as a strong reference model for EvoCON. It performs retrieval in a shared vision–language embedding space with frozen multimodal embeddings, lightweight script-conditioned adaptation, and a multi-prototype dictionary, allowing us to analyze how textual grounding, modular adaptation, and prototype construction affect continual glyph retrieval.

Our main contributions are as follows:

- We formulate ancient CCR as *language-grounded continual glyph lexicon expansion*, shifting the task from closed-set classification to expandable dictionary retrieval over script-aware lexical entries.
- We introduce **EvoCON**, a language-grounded benchmark with six-stage script onboarding, meaning and shape descriptions, and a zero-shot candidate retrieval track.
- We provide **AMR-CCR** as a strong reference model and use it to analyze the roles of textual grounding, script-conditioned adaptation, and multi-prototype dictionary construction.

2 Related Work

2.1 Ancient Character Recognition

Ancient character recognition is an important component of cultural-heritage digitization, with progress driven by both glyph resources and recognition models (Li et al., 2024; Wang et al., 2024b; Wang and Deng, 2022; Huang et al., 2019). For ancient Chinese Character Recognition (CCR), datasets such as OBC306, Oracle-MNIST, and EVOBC provide large-scale glyph images and labels for studying oracle-bone and multi-script recognition (Huang et al., 2019; Wang and Deng, 2022; Guan et al., 2024), while complementary resources support oracle-bone recognition and decipherment beyond a fixed vocabulary (Wang et al., 2024a). Most existing CCR methods, however, are still formulated as static closed-set recognition, improving robustness through stronger visual backbones, component-aware modeling, data homogenization, augmentation, or noise-robust learning (Wang et al., 2026, 2025, 2023). In contrast, our work studies ancient CCR as an expandable language-grounded lexicon: EvoCON augments glyph images with script-aware labels, meaning descriptions, shape descriptions, and zero-shot candidate retrieval, enabling evaluation beyond fixed-label classification.

2.2 Continual Learning for Recognition and Retrieval

Continual learning is commonly studied in class-incremental recognition, where models learn new classes or tasks sequentially while preserving old knowledge. Representative approaches include rehearsal, regularization, distillation, exemplar-based learning, and replay-based baselines (Chaudhry et al., 2019; Kirkpatrick et al., 2017; Rebuffi et al., 2017; Buzzega et al., 2020). A few works have

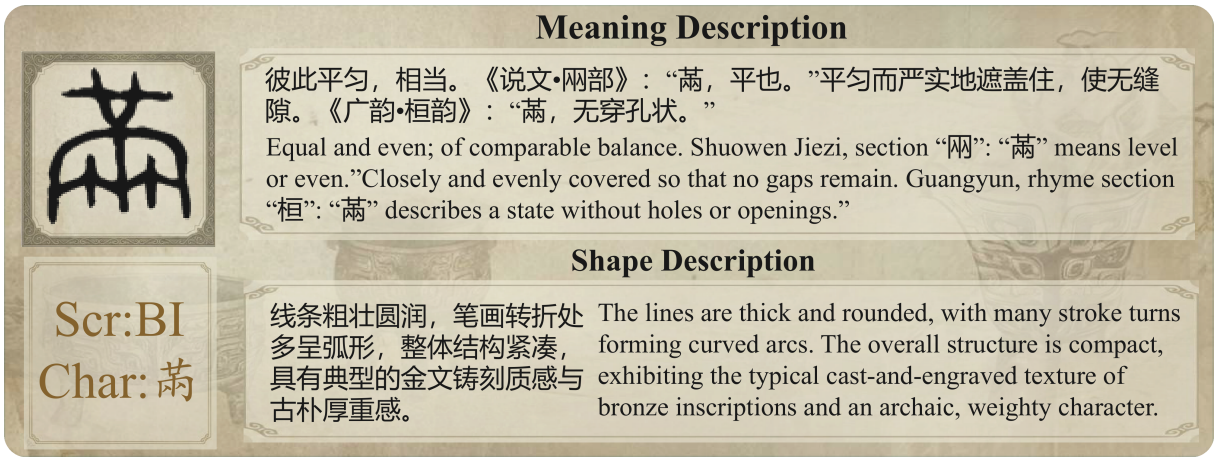


Figure 2: An EvoCON example with image, `scr` (script label), `char` (modern character label), and paired textual annotations: a character-level meaning description and an image-level shape description.

explored continual learning for ancient CCR to accommodate newly arriving scripts, samples, or categories (Xu et al., 2024), but they are typically classifier-centric. Our setting is retrieval-oriented: predictions depend on similarity to stored dictionary entries rather than only on classifier outputs. This makes cross-stage embedding compatibility crucial, since representation drift can make previously stored features or prototypes incompatible with the current embedding space (Yu et al., 2020; Goswami et al., 2025). We study this continual retrieval problem in the fine-grained ancient glyph domain, where the lexicon expands by script and each class may contain diverse visual styles.

2.3 Vision–Language Embeddings for Language-Grounded Retrieval

Vision–language contrastive pretraining, such as CLIP and ALIGN, learns shared image–text embeddings that support cross-modal retrieval and zero-shot recognition (Radford et al., 2021; Jia et al., 2021). Recent multimodal embedding models, including Qwen3-VL and Qwen3-VL-Embedding, further provide retrieval-oriented representations for unified image–text similarity search (Bai et al., 2025; Li et al., 2026). These models are relevant to ancient CCR because textual knowledge naturally exists in dictionaries, paleographic descriptions, and scholarly annotations. However, most vision–language retrieval systems are evaluated with fixed candidate pools and static indexes. EvoCON instead studies language-grounded retrieval under a growing glyph lexicon, where textual descriptions serve not only as auxiliary supervision but also as lexical evidence for candidate retrieval. AMR-CCR provides a reference model for analyzing how such language grounding interacts with continual onboarding, script adaptation, and prototype-based dictionary construction.

3 EvoCON Benchmark

This section introduces **EvoCON**, the first language-grounded benchmark for continual ancient Chinese glyph retrieval. Built upon EVOBC (Guan et al., 2024), EvoCON extends image-label annotations with script-aware lexical entries, character-level meaning descriptions, and image-level shape descriptions. It supports two tracks: continual onboarding, where the searchable glyph lexicon expands by script, and zero-shot candidate retrieval, where held-out query images are matched against textual candidate entries when no image exemplars are available.

3.1 Task Definition

We formulate ancient CCR as language-grounded continual glyph lexicon retrieval. A lexical entry is defined as $e = (s, c)$, where s denotes the script identity and c denotes the modern character identity. We use script-aware entries because the same modern character may undergo systematic visual transformations across scripts; collapsing them into a single class would mix cross-script evolution with ordinary within-class variation.

Each entry is grounded by visual and textual evidence. When image exemplars are available, an entry can be represented by glyph images or image-derived prototypes. Textually, each modern character is associated with a meaning description, and each observed glyph instance is associated

Table 1: Statistics of EvoCON splits across scripts. Img: number of images; Ent.: number of script-aware lexical entries; MCL: mean length of meaning captions; SCL: mean length of shape captions. The dash in the zero-shot SCL column indicates that shape descriptions are not provided for zero-shot queries.

	Train				Test				Zero-shot			
Script	Img	Ent.	MCL	SCL	Img	Ent.	MCL	SCL	Img	Ent.	MCL	SCL
OBC	67755	1632	284.0	47.2	7139	1555	298.4	46.8	413	104	314.8	–
BI	40448	2427	289.0	46.9	1886	883	319.6	45.7	379	134	271.7	–
SS	3881	3250	260.1	45.8	1561	1561	287.1	48.1	405	280	289.5	–
SAC	6961	1358	303.3	47.5	909	738	292.9	47.1	135	75	281.0	–
WSC	65136	3378	302.2	48.7	1239	937	287.1	48.1	442	180	324.3	–
CS	1885	1772	331.5	46.6	299	299	366.6	46.3	353	280	282.7	–
Total	186066	13817	295.0	47.1	13033	5973	308.6	47.0	2127	1053	294.0	–

with a shape description. At inference time, the query is always image-only: ground-truth script identity and query-side textual descriptions are not provided. The model outputs a ranked list of candidate lexical entries from the current searchable lexicon.

3.2 Dataset Construction

EvoCON is built from EVOBC, which contains glyph images and character labels across six historical scripts: Oracle Bone inscriptions (OBC), Bronze Inscriptions (BI), Seal Script (SS), Spring-and-Autumn Characters (SAC), Warring States Characters (WSC), and Clerical Script (CS). We retain the original glyph images and script metadata, convert labels into script-aware lexical entries, and further augment the data with language-grounded annotations that describe both character semantics and observable glyph forms.

Language-grounded annotations. EvoCON provides two types of Simplified Chinese textual annotations. *Meaning descriptions* are constructed at the character level by extracting dictionary meanings from an open-source Chinese lexicon and manually normalizing them for consistency and coverage. They are shared across scripts for the same modern character, serving as script-invariant semantic anchors. *shape descriptions* are constructed at the image-instance level by prompting Qwen3-VL-Plus (Bai et al., 2025) to describe observable glyph-form cues such as layout, component arrangement, stroke curvature, balance, and overall structure, followed by human verification and normalization. Details of data sources, prompting templates, and quality control are provided in Appendix A.

Splits and statistics. Since EVOBC does not provide an official train/test split, we construct script-wise splits for continual and zero-shot evaluation. For the continual track, each script is split so that test entries have image evidence in the corresponding training split. This design evaluates whether a model can maintain retrieval quality over an expanding observed lexicon, while leaving recognition of entries without image exemplars to the zero-shot track. For the zero-shot track, we hold out low-resource modern characters with fewer than five images in total across the six scripts, remove all images of their corresponding script-aware entries from training, and use them only for zero-shot evaluation.

As shown in Table 1, EvoCON contains 186,066 training images over 13,817 script-aware entries, 13,033 test images over 5,973 entries, and 2,127 zero-shot query images over 1,053 held-out entries. The splits are naturally long-tailed and imbalanced across scripts: OBC, BI, and WSC contribute most images, while scripts such as SS contain many entries but relatively few images. This reflects realistic variation in preservation quality, script coverage, and annotation density.

3.3 Continual Onboarding Protocol

The continual track evaluates whether a model can maintain retrieval quality as new scripts and lexical entries are incorporated over time. At each stage t, the learner receives the training split of one script and expands the searchable lexicon with the entries introduced at that stage. Future-stage data are strictly unavailable. By default, previous-stage

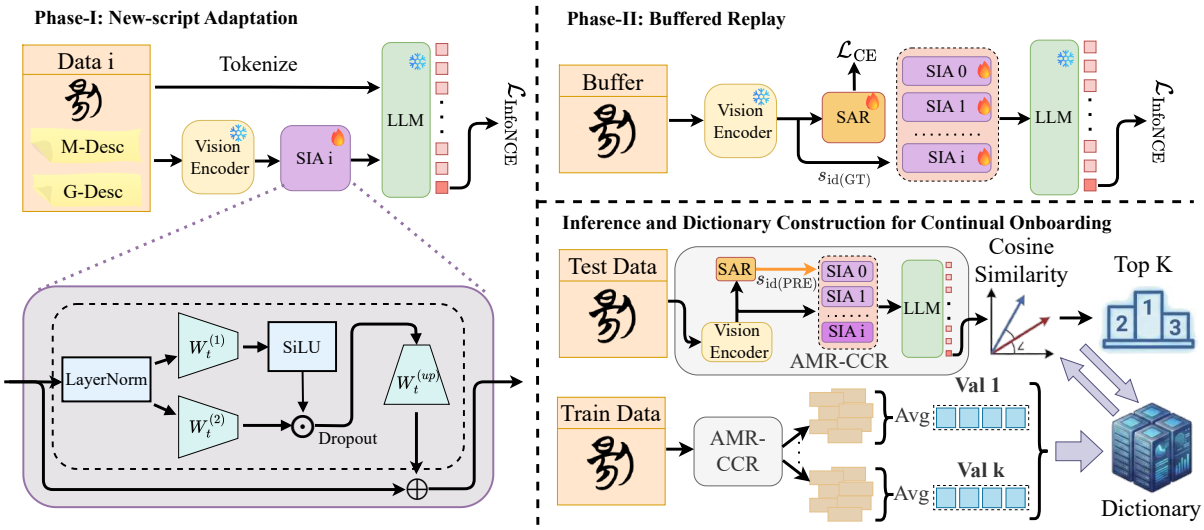


Figure 3: **Overview of AMR-CCR as a reference model for EvoCON.**

training data are not accessible, while rehearsal-based methods may retain a limited memory buffer.

We use the default six-stage sequence CS → WSC → SAC → SS → BI → OBC, following a reverse-chronological digitization scenario where more standardized scripts are integrated before earlier and more irregular scripts. After each stage, the model is evaluated on all observed scripts and must rank all observed script-aware entries. We report Top-1/Top-10 accuracy, stage-wise average accuracy AA1–AA6, and final-stage forgetting. Formal metric definitions are provided in Appendix C.

3.4 Zero-shot Candidate Retrieval

The zero-shot track evaluates whether a model can retrieve held-out lexical entries when no image exemplars of the target entries are available. This task models candidate verification with a provided lexicon or hypothesized candidate set, rather than unrestricted open-ended decipherment.

For each zero-shot query image, the target script-aware entry is excluded from all training data and has no image prototype in the dictionary. To keep the task consistent with script-aware recognition, each candidate is represented as $e = (s, c)$, and its textual entry is constructed by combining the script name s with the meaning description of the modern character c. This avoids ambiguity when the same modern character appears in multiple scripts with the same meaning description.

At inference, each query is image-only: ground-truth script identity, query-side meaning descriptions, and query-side shape descriptions are not provided. Unless otherwise specified, the candidate pool contains all 1,053 held-out zero-shot script-aware entries and is fixed across all methods. We report ZS@1 and ZS@20.

4 AMR-CCR Framework

To make EvoCON actionable and reproducible, we provide **AMR-CCR** as a strong reference model for language-grounded continual glyph retrieval. AMR-CCR is designed to test three practical ingredients for the proposed benchmark: a stable vision–language embedding space, lightweight script-conditioned adaptation, and an expandable dictionary for retrieval. It freezes a pretrained multimodal embedding backbone as a stability anchor, calibrates newly onboarded scripts with small script-specific modules, and represents each lexical entry with either image-derived prototypes or textual candidate entries.

4.1 Script-Conditioned Embedding

We adopt Qwen3-VL-Embedding (Li et al., 2026) as the shared vision–language backbone. Given an image x, the frozen visual encoder produces an embedding $\mathbf{e} = E_{\text{vis}}(x) \in R^D$. For each onboarded script t, AMR-CCR maintains a lightweight Script-Interface Adapter (SIA), denoted by $\phi_t(\cdot)$, which applies a script-conditioned low-rank residual correction:

$$\mathbf{h} = \text{LN}(\mathbf{e}), \quad \mathbf{a}_t = W_t^{(1)}\mathbf{h}, \quad \mathbf{b}_t = W_t^{(2)}\mathbf{h}. \quad (1)$$

$$\mathbf{z}_t = \sigma(\mathbf{a}_t) \odot \mathbf{b}_t. \quad (2)$$

$$\phi_t(\mathbf{e}) = norm(\mathbf{e} + \alpha_t W_t^u \mathbf{z}_t)\,. \quad (3)$$

Here $W_t^{(1)}, W_t^{(2)} \in R^{r\times D}, W_t^u \in R^{D\times r}$, and $r \ll D$. This adapter keeps the update small and script-specific while preserving a shared retrieval space.

Since the query script is not provided at inference, AMR-CCR uses a Script-Aware Router (SAR) to select the adapter. Given the frozen embedding $\mathbf{e}$, the router predicts an adapter distribution and selects one path:

$$\mathbf{p} = g(\mathbf{e}) \in R^t, \qquad k = \arg\max_j \mathbf{p}_j. \quad (4)$$

The routed query embedding is then $\mathbf{q} = \phi_k(\mathbf{e})$. SAR only selects the script-conditioned calibration path; the final character decision is made by dictionary retrieval rather than by the router.

4.2 Stage-wise Training

AMR-CCR follows the EvoCON onboarding protocol. At each stage t, the pretrained backbone remains frozen, and training is performed in two phases: new-script adaptation and buffered replay.

Phase-I: New-Script Adaptation. For the newly onboarded script, we freeze the backbone, the router, and all previous adapters, and optimize only the new adapter $\phi_t(\cdot)$. Each training image is used as an anchor, and its positive target is sampled from one of three sources: another image of the same script-aware lexical entry, the character-level meaning description, or the image-level shape description. By default, visual, meaning, and shape positives are sampled with a ratio of 8:1:1.

We use the same contrastive objective for image–image and image–text pairs. For anchor embedding $\mathbf{a}_i$, candidate embedding $\mathbf{u}$, and temperature τ, define

$$s_i(\mathbf{u}) = \exp(\mathrm{sim}(\mathbf{a}_i, \mathbf{u})/\tau)\,. \quad (5)$$

The loss for the sampled positive $\mathbf{u}_i$ is

$$\ell_i = -\log \frac{s_i(\mathbf{u}_i)}{\sum_{\mathbf{u}\in\mathcal{U}_i} s_i(\mathbf{u})}, \quad (6)$$

where $\mathcal{U}_i$ is the in-batch candidate set and $\mathrm{sim}(\cdot,\cdot)$ denotes cosine similarity on normalized embeddings. In Phase-I, $\mathbf{a}_i = \phi_t(E_{\text{vis}}(x_i))$, while $\mathbf{u}_i$ is the embedding of the sampled image or text positive.

Phase-II: Buffered Replay. After new-script adaptation, we use a fixed memory buffer of 10,000 images, sampled approximately evenly from all observed scripts. This phase has two roles: realigning the adapter bank across scripts and training SAR for inference-time routing. For each buffered pair (x_i, x_i^+) from the same script-aware lexical entry, we compute embeddings with the corresponding script adapter and apply the contrastive loss in Eq. 6. In parallel, the router is trained with script labels:

$$\mathcal{L}_{\text{route}} = -\log g(E_{\text{vis}}(x_i))_{s_i}. \quad (7)$$

During Phase-II, the contrastive loss updates only the SIA bank, while $\mathcal{L}_{\text{route}}$ updates only the router. The pretrained vision–language backbone remains frozen throughout all stages.

4.3 Dictionary Construction and Inference

AMR-CCR performs recognition by ranking dictionary entries in the shared embedding space. The continual and zero-shot tracks differ only in how the dictionary entries are represented.

Observed-entry dictionary. For the continual track, each observed lexical entry y is represented by a set of image-derived prototypes. We encode its training images with the corresponding routed adapter and cluster the embeddings using spherical k-means:

$$\mathcal{P}_y = \{\mathbf{c}_{y,1}, \ldots, \mathbf{c}_{y,K_y}\}. \quad (8)$$

The number of prototypes K_y is selected automatically using a silhouette criterion on a capped subset of embeddings, with $K_y \le K_{\max}$. Entries with too few examples use a single prototype. The full Auto-K procedure is provided in Appendix D.1.

Given an image-only query x, let $\mathbf{e} = E_{\text{vis}}(x)$. AMR-CCR computes

$$k = \arg\max_j g(\mathbf{e})_j, \qquad \mathbf{q} = \phi_k(\mathbf{e}). \quad (9)$$

The score of an observed lexical entry is the maximum cosine similarity over its prototypes:

$$S(y \mid x) = \max_{\mathbf{c}\in\mathcal{P}_y} \mathbf{q}^\top \mathbf{c}. \quad (10)$$

All observed entries are ranked by $S(y \mid x)$ to produce Top-K predictions.

Zero-shot textual dictionary. For the zero-shot candidate retrieval track, held-out entries have no image prototypes. Following the EvoCON protocol, each candidate entry $e = (s, c)$ is represented by a script-aware textual description constructed

Table 2: Results under the six-stage continual onboarding protocol. Each entry reports Top-1/Top-10 accuracy. Seq FT and Joint FT denote sequential and joint fine-tuning, respectively. Frozen and Joint FT yield a single final model, so only AA6 is reported.

Method	**AA1** ↑	**AA2** ↑	**AA3** ↑	**AA4** ↑	**AA5** ↑	**AA6** ↑	**FGT** ↓
ResNet-50 (He et al., 2016) Based							
Seq FT	0.00/3.34	8.64/20.37	1.98/57.57	0.73/1.40	12.17/13.04	14.69/16.34	16.81/27.89
Joint FT	–	–	–	–	–	42.38/62.67	–
Qwen3-VL-Embedding-8B (Li et al., 2026) Based							
Frozen	–	–	–	–	–	36.68/55.83	–
Joint FT	–	–	–	–	–	41.14/66.93	–
Seq FT	67.22/87.62	24.13/43.73	21.63/48.34	22.91/49.06	16.53/38.10	28.68/55.25	9.52/9.70
ER (Wu et al., 2019)	66.88/86.62	36.01/54.45	29.40/50.55	27.74/52.40	26.29/49.51	34.74/59.09	3.09/4.44
LwF (Li and Hoiem, 2017)	68.18/88.20	38.50/56.95	31.04/53.17	30.32/54.98	28.78/52.00	36.82/61.13	2.81/3.65
EWC (Kirkpatrick et al., 2017)	68.43/88.79	38.85/56.85	32.03/52.93	30.65/55.15	29.10/52.28	37.26/61.49	2.94/3.71
iCaRL (Rebuffi et al., 2017)	66.22/86.28	36.46/52.24	30.64/49.69	28.35/51.26	25.84/49.10	31.74/56.15	2.97/3.34
DER++ (Buzzega et al., 2020)	**68.67**/88.43	40.40/64.11	35.51/59.13	40.80/64.39	42.05/64.80	45.85/68.01	2.54/3.15
AMR-CCR (2B)	57.86/84.95	39.52/65.97	37.61/63.56	46.12/70.62	46.87/70.37	50.43/72.86	**1.16/1.70**
AMR-CCR (8B)	68.40/**90.02**	**55.49/84.60**	**50.67/80.29**	**56.12/83.61**	**57.06/81.82**	**58.59/83.09**	1.85/2.66

from the script name s and the meaning description of the modern character c. Let $\mathbf{t}_e$ denote its normalized text embedding. The zero-shot score is

$$S(e \mid x) = \mathbf{q}^\top \mathbf{t}_e. \tag{11}$$

The model ranks all candidate entries by this image–text similarity and reports ZS@1 and ZS@20.

5 Experiments

5.1 Experimental Setup

Evaluation tracks and metrics. All experiments are conducted on EvoCON following the benchmark protocol in Sec. 3. We evaluate two tracks. For the continual onboarding track, after each stage t, we evaluate on all observed scripts and report Top-1/Top-10 retrieval accuracy. We summarize the stage-wise trajectory with average accuracy AA1–AA6 and final-stage forgetting (FGT). For the zero-shot candidate retrieval track, target entries are excluded from training and have no image prototypes; the model ranks textual candidate entries and we report ZS@1 and ZS@20. Formal metric definitions are provided in Appendix C.

Compared methods. We compare three groups of methods. The first group contains ResNet-50 classification baselines, including sequential fine-tuning and joint training. The second group contains Qwen3-VL-Embedding retrieval baselines, including the frozen pretrained model, joint fine-tuning, sequential fine-tuning, and representative continual-learning methods: ER, LwF, EWC, iCaRL, and DER++. These baselines cover rehearsal, regularization/distillation, exemplar-based learning, and replay with distillation. The third group contains AMR-CCR variants, which serve as reference systems for analyzing EvoCON.

Implementation details. All methods are implemented in PyTorch with mixed precision. AMR-CCR uses Qwen3-VL-Embedding-2B and 8B (Li et al., 2026) as frozen backbones. Unless otherwise stated, the SIA bottleneck rank is r=64, the residual scale is initialized as α=0.5, and the replay memory stores 10,000 images with approximately balanced script sampling. We train with AdamW using learning rate 1e−4, weight decay 0.1, cosine scheduling, warmup ratio 0.01, batch size 128, and 6/5 epochs for Phase-I/II. InfoNCE uses temperature τ=0.1. For dictionary retrieval, we use ℓ_2 normalization and cosine similarity. The Auto-K dictionary uses $K_{\max}$=32 and per-entry sample cap S=256. Baseline reproduction details are provided in Appendix E.

5.2 Continual Glyph Retrieval

Table 2 shows that EvoCON is challenging for both classification-based and embedding-based continual learning. With a ResNet-50 classifier, sequential fine-tuning reaches only 14.69/16.34 at AA6 and suffers from severe forgetting, while joint training improves to 42.38/62.67. This gap indicates that continual glyph lexicon expansion is difficult not only because of the visual backbone, but also because the label space is long-tailed, fine-grained,

Table 3: Diagnostic results under the continual protocol and the zero-shot (ZS) track. Continual results report final-stage Top-1/Top-10; ZS reports Top-1/Top-20. Gold denotes ground-truth script routing, and RS denotes random sampling.

Variant	**AA6** ↑	**FGT** ↓	**ZS** ↑
AMR-CCR (8B)	**58.6/83.1**	1.9/2.7	15.2/46.8
AMR-CCR (2B)	50.4/72.9	**1.2/1.7**	**20.9**/38.6
w/o SIA & SAR	28.7/55.3	9.5/9.7	–
w/o SAR (Gold)	61.2/86.8	0.9/1.7	15.9/**48.8**
Buffer 5k	57.2/80.8	2.8/3.6	14.3/43.9
Buffer 15k	59.1/83.8	1.7/2.4	15.2/47.0
Mean prototype	48.5/70.2	1.6/2.2	–
RS (M=8)	46.4/65.4	1.6/2.1	–
Image-only	43.4/65.8	1.1/1.9	9.9/25.9

and continually expanding.

Qwen3-VL-Embedding provides stronger pretrained representations, but naive continual adaptation remains unstable. The frozen 8B model obtains 36.68/55.83 at AA6, and joint fine-tuning improves to 41.14/66.93. However, sequential fine-tuning drops to 28.68/55.25 with FGT 9.52/9.70, suggesting that stage-wise updates can break cross-stage similarity compatibility. Standard continual learning methods reduce forgetting, but their final retrieval accuracy remains limited; DER++ is the strongest baseline, reaching 45.85/68.01 at AA6.

AMR-CCR achieves the best final retrieval performance among the compared methods. The 8B reference model reaches 58.59/83.09 at AA6, outperforming DER++ by 12.74/15.08 while keeping forgetting low at 1.85/2.66. The 2B variant also outperforms DER++ in final AA6, reaching 50.43/72.86 with even lower forgetting. These results support the suitability of frozen multimodal embeddings, script-conditioned lightweight adaptation, and explicit dictionary construction for continual glyph lexicon expansion.

5.3 Diagnostic Analysis

Table 3 summarizes the main diagnostic results; full stage-wise trajectories are provided in Appendix F.1. Scaling from 2B to 8B improves continual retrieval accuracy (50.4/72.9 → 58.6/83.1), showing that larger multimodal embeddings better separate near-duplicate entries. The 2B model has lower forgetting and higher ZS@1, while 8B achieves higher ZS@20, suggesting a trade-off between stability, top-rank calibration, and candidate recall.

Script-conditioned adaptation is critical for continual onboarding. Removing SIA and SAR reduces AA6 to 28.7/55.3 and increases forgetting to 9.5/9.7, close to naive sequential fine-tuning. This shows that simply updating a shared embedding model is insufficient for maintaining cross-stage compatibility. With ground-truth script routing, performance rises to 61.2/86.8, providing an oracle upper bound. The full model remains close to this bound, suggesting that SAR handles most script-level routing decisions and that many remaining errors come from fine-grained character confusion.

Replay and prototype construction address different bottlenecks. Increasing the buffer from 5k to 15k improves AA6 from 57.2/80.8 to 59.1/83.8 and reduces forgetting, though the gain over the default 10k buffer is modest. Replacing Auto-K multi-prototypes with a mean prototype or random sampling lowers AA6 to 48.5/70.2 and 46.4/65.4, showing that multi-prototype retrieval is important for covering diverse writing styles within each lexical entry.

Language grounding also matters. Removing textual supervision from Phase-I drops AA6 from 58.6/83.1 to 43.4/65.8 and reduces ZS from 15.2/46.8 to 9.9/25.9. This shows that textual descriptions provide useful supervision beyond image-only contrastive training, especially for image-to-text candidate retrieval in the zero-shot track. In our setting, textual supervision consists of character-level meaning descriptions and image-level shape descriptions.

6 Conclusion

We formulated ancient Chinese character recognition as *language-grounded continual glyph lexicon retrieval*, shifting the task from closed-set classification to retrieval over an expandable dictionary of script-aware lexical entries. We introduced **EvoCON**, a benchmark with multi-script glyph images, meaning and shape descriptions, continual onboarding, and zero-shot candidate retrieval, and provided **AMR-CCR** as a strong reference model for analysis. Experiments show that EvoCON exposes challenges hidden by static closed-set evaluation, including continual lexicon growth, fine-grained inter-entry confusion, and intra-entry style variation. Our results suggest that language-grounded dictionary retrieval is a practical paradigm for continually expanding ancient glyph lexicons.

Limitations

This work is an initial exploration of ancient CCR under a setting that better matches real digitization workflows, where the glyph lexicon expands over time and recognition relies on both visual evidence and textual knowledge. EvoCON provides a controlled benchmark for this setting, and our experiments provide preliminary evidence for the value of language-grounded dictionary retrieval. However, EvoCON currently covers six historical Chinese scripts, and its applicability to broader writing systems remains to be validated. The zero-shot track should also be interpreted as candidate retrieval rather than unrestricted decipherment. It evaluates whether a held-out glyph image can be matched to a provided textual candidate lexicon when no image prototype is available, but it does not solve the harder case where no reliable candidate set or textual description exists.

References

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, and 1 others. 2025. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. 2020. Dark experience for general continual learning: a strong, simple baseline. *Preprint*, arXiv:2004.07211.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K. Dokania, Philip H. S. Torr, and Marc'Aurelio Ranzato. 2019. On tiny episodic memories in continual learning. *Preprint*, arXiv:1902.10486.

Raúl Fuentes-Ferrer, Jaime Duque-Domingo, and Pedro Javier Herrera. 2025. Recognition of egyptian hieroglyphic texts through focused generic segmentation and cross-validation voting. *Applied Soft Computing*, 171:112793.

Shai Gordin, Morris Alper, Avital Romach, Luis Saenz Santos, Naama Yochai, and Roey Lalazar. 2024. Cured: Deep learning optical character recognition for cuneiform text editions and legacy materials. In *Proceedings of the 1st Workshop on Machine Learning for Ancient Languages (ML4AL 2024)*, pages 130–140.

Dipam Goswami, Liying Wang, BartĹ Twardowski, Joost van de Weijer, and 1 others. 2025. Query drift compensation: Enabling compatibility in continual learning of retrieval embedding models. *arXiv preprint arXiv:2506.00037*.

Haisu Guan, Jinpeng Wan, Yuliang Liu, Pengjie Wang, Kaile Zhang, Zhebin Kuang, Xinyu Wang, Xiang Bai, and Lianwen Jin. 2024. An open dataset for the evolution of oracle bone characters: Evobc. *Preprint*, arXiv:2401.12467.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Shuangping Huang, Haobin Wang, Yongge Liu, Xiaosong Shi, and Lianwen Jin. 2019. Obc306: A large-scale oracle bone character recognition dataset. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 681–688. IEEE.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pages 4904–4916. PMLR.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Jing Li, Xueke Chi, Qiufeng Wang, Dahan Wang, Kaizhu Huang, Yongge Liu, and Cheng lin Liu. 2024. A comprehensive survey of oracle character recognition: challenges, benchmarks, and beyond. *Preprint*, arXiv:2411.11354.

Mingxin Li, Yanzhao Zhang, Dingkun Long, Keqin Chen, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2026. Qwen3-vl-embedding and qwen3-vl-reranker: A unified framework for state-of-the-art multimodal retrieval and ranking. *arXiv*.

Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting. *Preprint*, arXiv:1606.09282.

Natalia Hernández Mejías. 2025. Technical guidelines for digitizing cultural heritage materials [guía técnica para la digitalización de materiales de herencia cultural. *Acceso. Revista Puertorriqueña de Bibliotecología y Documentación*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and 1 others. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. 2017. icarl:

Incremental classifier and representation learning. *Preprint*, arXiv:1611.07725.

Matthew I Swindall, Gregory Croisdale, Chase C Hunter, Ben Keener, Alex C Williams, James H Brusuelas, Nita Krevans, Melissa Sellew, Lucy Fortson, and John F Wallin. 2021. Exploring learning approaches for ancient greek character recognition with citizen science data. In *2021 IEEE 17th International conference on eScience (eScience)*, pages 128–137. IEEE.

Kaili Wang, Tianquan Wu, Yuanlin Shi, and Chen Chen. 2026. Attgraph disentangling confusable ancient chinese characters via component-correlation synergy. *npj Heritage Science*, 14.

Mei Wang and Weihong Deng. 2022. Oracle-mnist: a dataset of oracle characters for benchmarking machine learning algorithms. *arXiv preprint arXiv:2205.09442*.

Nan Wang, Weichen Wang, Bang Li, Han Zhang, Qingju Jiao, and Chaofan Liu. 2025. Multi-modal ancient scripts recognition via deep learning with data homogenization and augmentation. *npj Heritage Science*, 13(1):522.

Pengjie Wang, Kaile Zhang, Xinyu Wang, Shengwei Han, Yongge Liu, Jinpeng Wan, Haisu Guan, Zhebin Kuang, Lianwen Jin, Xiang Bai, and Yuliang Liu. 2024a. An open dataset for oracle bone script recognition and decipherment. *Preprint*, arXiv:2401.15365.

Pengjie Wang, Kaile Zhang, Xinyu Wang, Shengwei Han, Yongge Liu, Jinpeng Wan, Haisu Guan, Zhebin Kuang, Lianwen Jin, Xiang Bai, and 1 others. 2024b. An open dataset for oracle bone character recognition and decipherment. *Scientific Data*, 11(1):976.

Xuanhong Wang, Kun Wu, Ying Zhang, Yun Xiao, and Pengfei Xu. 2023. A gan-based denoising method for chinese stele and rubbing calligraphic image. *The visual computer*, 39(4):1351–1362.

Chenshen Wu, Luis Herranz, Xialei Liu, Yaxing Wang, Joost van de Weijer, and Bogdan Raducanu. 2019. Memory replay gans: learning to generate images from new categories without forgetting. *Preprint*, arXiv:1809.02058.

Yue Xu, Xu-Yao Zhang, Zhaoxiang Zhang, and Cheng-Lin Liu. 2024. Large-scale continual learning for ancient chinese character recognition. *Pattern Recognition*, 150:110283.

Lu Yu, Bartłomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. 2020. Semantic drift compensation for class-incremental learning. *Preprint*, arXiv:2004.00440.

A EvoCON Construction Details

This section provides additional details on the construction of EvoCON, including source preprocessing, language-grounded annotation, prompting, and human verification.

A.1 Source Dataset and Preprocessing

EvoCON is built upon EVOBC and covers six historical scripts: Oracle Bone inscriptions (OBC), Bronze Inscriptions (BI), Seal Script (SS), Spring-and-Autumn Characters (SAC), Warring States Characters (WSC), and Clerical Script (CS). For each sample, we retain the original glyph image, the script metadata, and the modern-character identity. We then convert the original labels into script-aware lexical entries $e = (s, c)$, where s denotes the script identity and c denotes the modern-character identity.

This design follows the benchmark definition in the main paper: the same modern character appearing in different scripts is treated as a different lexical entry, because cross-script transformations reflect systematic historical evolution rather than ordinary within-entry variation. We perform basic preprocessing to unify the data format across scripts, including image normalization, metadata cleaning, and label consistency checking. After preprocessing, each observed sample is represented as

$$(x, s, c, e, t^{\text{mean}}, t^{\text{form}}),$$

where x is the glyph image, s is the script identity, c is the modern-character identity, $e = (s, c)$ is the script-aware lexical entry, t^{mean} is the character-level meaning description, and t^{form} is the image-level shape description.

A.2 Meaning Description Construction

To provide script-shared semantic supervision, we associate each modern character with a meaning description. These descriptions are intended to capture the core semantics of the corresponding modern character.

Specifically, we first collect candidate semantic descriptions from dictionary entries on the Han Dian website.[1] The resulting text is manually normalized to improve consistency and coverage. This normalization process includes removing overly long or redundant expressions, unifying terminology, resolving obvious ambiguity when possible,

[1] https://www.zdic.net/

and rewriting the descriptions into a concise declarative format suitable for text encoding.

When multiple scripts correspond to the same modern character, we use the same meaning description across scripts. This design encourages the model to align visually different script forms through shared semantics rather than relying only on direct visual similarity.

A.3 Shape Description Construction

In addition to semantic text, EvoCON provides image-level shape descriptions that characterize observable glyph-form cues. Unlike meaning descriptions, shape descriptions are tied to individual glyph images and may capture script style, global layout, component arrangement, stroke configuration, curvature, balance, and local structure.

We generate initial shape descriptions using Qwen3-VL-Plus given the input glyph image and script category. The model is prompted to describe only directly observable visual features and to avoid semantic interpretation, pronunciation, radicals, stroke counts, or guesses about the corresponding modern character. Figure 4 shows the prompt template used for shape description generation.

After generation, raw outputs are further reviewed and normalized by human annotators. This post-processing stage removes hallucinated semantic content, deletes repetitive phrases, unifies descriptive style, and corrects visually inconsistent statements. We use the term *shape description* rather than *shape description* to emphasize that these annotations may contain a mixture of script-level style cues and glyph-specific form cues.

A.4 Quality Control and Human Verification

We apply quality control to both meaning and shape descriptions. For meaning descriptions, we verify whether the text faithfully reflects the intended semantics of the modern character and whether the wording is concise and unambiguous. For shape descriptions, we verify whether the generated text is consistent with the input image and whether it focuses on observable form rather than semantic content.

Human verification is performed to filter or revise low-quality annotations. Typical issues include: (i) semantic mismatch, (ii) visually inaccurate description, (iii) overly generic text with limited discriminative value, (iv) leakage from modern-character priors that is not visually grounded, and (v) repetitive or stylistically inconsistent expressions. When an annotation fails verification, it is either manually revised or discarded.

A.5 Zero-shot Candidate Entry Construction

The zero-shot track evaluates candidate retrieval rather than unrestricted open-ended decipherment. The model is given a fixed candidate lexicon and must rank textual candidate entries for an image-only query.

For each held-out lexical entry $e = (s, c)$, we construct a script-aware textual candidate entry by combining the script name s and the meaning description of the modern character c. This avoids ambiguity when the same modern character appears in multiple scripts with the same meaning description. The candidate template is:

```
This is a [SCRIPT] glyph.
Its modern character meaning is:
[MEANING].
```

In Chinese, the corresponding template is:

```
[SCRIPT][MEANING]
```

Unless otherwise specified, the zero-shot candidate pool contains all 1,053 held-out script-aware lexical entries. Query-side script labels, meaning descriptions, and shape descriptions are not provided at inference.

B Additional EvoCON Examples and Annotation Analysis

This section provides additional visualized examples and further analysis of the language-grounded annotations in EvoCON.

B.1 More Visualized Examples

Figures 5–9 show representative examples from EvoCON. Each example includes a glyph image, script identity, modern-character identity, meaning description, and shape description.

B.2 Analysis of Meaning and shape descriptions

Meaning and shape descriptions serve different roles. Meaning descriptions are associated with modern characters and are shared across scripts, so they mainly provide script-invariant semantic anchors. shape descriptions are image-grounded and vary with the observed glyph instance, so they capture observable form cues such as script style, layout, stroke configuration, and local structure.

Prompt Template

You are generating a training caption for an ancient character image.

Input:
- Script category: {script_category}
- Image: <IMAGE>

Task:
Write a concise Chinese description of the glyph shown in the image.

Output requirements:
- Output exactly one paragraph in Chinese.
- The paragraph must begin with "字形分析：".
- Length: 20–35 Chinese characters (including the prefix if counted by the system).
- Describe only directly observable visual features of the glyph, such as overall shape, stroke style, structure, curvature, balance, density, or layout.
- The description must be consistent with the given script category.
- Use specific, visible details rather than vague praise or generic wording.

Restrictions:
- Do not mention meaning, pronunciation, radicals, or stroke count.
- Do not guess the corresponding modern character.
- Do not cite or invent any source, book, dictionary, or encyclopedia information.
- Do not include explanations, labels, or extra text beyond the single required paragraph.

Figure 4: Prompt used in Qwen3-VL-Plus for initial shape description generation. The model is instructed to describe observable glyph-form cues rather than semantic interpretation.

We emphasize that shape descriptions are not assumed to be purely character-discriminative. They may encode a mixture of script-level style and glyph-specific form cues. This makes them useful as a form of language grounding, but also suggests that future work may further disentangle semantic, script-style, and glyph-specific textual supervision.

C Formal Metric Definitions

This section presents the formal definitions of all evaluation metrics used in the paper. We consider a $T = 6$ stage continual retrieval setting, where each target is a script-aware lexical entry. Let f_t denote the model after training up to stage t, and let $\mathcal{D}_i^{te}$ be the test split of stage i. We use $\mathrm{Acc}_{t,i}^{@k}$ to denote the Top-k retrieval accuracy of f_t evaluated on $\mathcal{D}_i^{te}$.

Stage-wise average accuracy. For each stage t, we evaluate f_t on all observed stages $\{1, \ldots, t\}$ and compute the stage-wise average accuracy:

$$\mathrm{AA}_t^{@k} = \frac{1}{t} \sum_{i=1}^{t} \mathrm{Acc}_{t,i}^{@k}. \tag{12}$$

We report AA1–AA6 for the continual track using $k \in \{1, 10\}$.

Forgetting. For a previously learned stage $i < t$, forgetting at stage t is defined as the drop from its best past performance to the current performance:

$$\mathrm{Fgt}_{t,i}^{@k} = \max_{\tau \in [i, t-1]} \mathrm{Acc}_{\tau,i}^{@k} - \mathrm{Acc}_{t,i}^{@k}. \tag{13}$$

The final-stage forgetting is

$$\mathrm{FGT}_T^{@k} = \frac{1}{T-1} \sum_{i=1}^{T-1} \mathrm{Fgt}_{T,i}^{@k}. \tag{14}$$

We report $\mathrm{FGT}_T^{@k}$ at the final stage $T = 6$ for $k \in \{1, 10\}$.

Zero-shot candidate retrieval. Let $\mathcal{D}^{zs}$ be the zero-shot query set and $\mathcal{C}^{zs}$ be the fixed textual candidate pool. We report:

$$\mathrm{ZS}^{@k} = \frac{1}{|\mathcal{D}^{zs}|} \sum_{(x,e)\in\mathcal{D}^{zs}} I[e \in \mathrm{Top}{-}k(f_T(x), \mathcal{C}^{zs})] \,. \tag{15}$$

We use $k \in \{1, 20\}$.

D Reference Model and Dictionary Details

This section provides additional details for AMR-CCR, the reference model used in the main paper.

D.1 Auto-K Multi-Prototype Dictionary

For each observed lexical entry y, AMR-CCR constructs an image-derived prototype set $\mathcal{P}_y$ from training embeddings. We use spherical k-means to better cover multiple writing styles within the same entry. The number of prototypes is selected automatically by silhouette score on a capped subset of embeddings.

For each entry y, let

$$\mathbf{E}_y = \{\mathbf{e}_i : y_i = y\}$$

be the set of training embeddings belonging to y. We set

$$K_y^{\max} = \min(K_{\max}, \lfloor\sqrt{|\mathbf{E}_y|}\rfloor).$$

If $K_y^{\max} < 2$, we use a single prototype. Otherwise, we sample a capped subset $\tilde{\mathbf{E}}_y \subseteq \mathbf{E}_y$ with $|\tilde{\mathbf{E}}_y| = \min(|\mathbf{E}_y|, S)$ and select

$$K_y = \arg\max_{k\in[2,K_y^{\max}]} \mathrm{Sil}_y(k),$$

where $\mathrm{Sil}_y(k)$ denotes the cosine silhouette score of spherical k-means on the capped subset $\tilde{\mathbf{E}}_y$. Finally, we run spherical k-means on the full embedding set $\mathbf{E}_y$ with K_y clusters and use the resulting centroids as $\mathcal{P}_y$. Algorithm 1 summarizes the full implementation.

D.2 Dictionary Scoring

For the continual onboarding track, the score of an observed lexical entry y for query image x is

$$S(y \mid x) = \max_{\mathbf{c}\in\mathcal{P}_y} \mathbf{q}^\top \mathbf{c},$$

where $\mathbf{q}$ is the routed query embedding and $\mathcal{P}_y$ is the prototype set of entry y.

For the zero-shot candidate retrieval track, each held-out entry e is represented by a normalized text embedding $\mathbf{t}_e$. The score is

$$S(e \mid x) = \mathbf{q}^\top \mathbf{t}_e.$$

E Baseline Reproduction Details

This section provides implementation details for the compared baselines. All methods are evaluated under the same EvoCON splits, stage order, candidate pools, and metrics.

E.1 ResNet-Based Classification Baselines

Seq FT. Seq FT is implemented as standard sequential supervised fine-tuning. The model is trained on the six stages in order, using only the training data from the current stage at each step. During training, the entire ResNet backbone is fully unfrozen. We use a learning rate of 5×10^{-5} and train for 6 epochs per stage.

Joint FT. Joint FT is implemented as a non-continual reference baseline. We combine the training data from all six stages into a single dataset and train once on the merged training set. The entire ResNet backbone is fully unfrozen. We use a learning rate of 5×10^{-5} and train for 6 epochs.

E.2 Qwen3-VL-Embedding Retrieval Baselines

This group includes baselines based on Qwen3-VL-Embedding, where recognition is performed through embedding extraction and similarity matching rather than a conventional classification head.

Frozen. The frozen baseline directly uses Qwen3-VL-Embedding without any EvoCON training. Image queries and dictionary entries are encoded with the frozen model, and candidates are ranked by cosine similarity.

Seq FT. Seq FT is implemented as sequential supervised fine-tuning under the same continual stage order as EvoCON. At each stage, the model is trained only on the training data of the current stage. We adopt LoRA fine-tuning and update only the LLM part of Qwen3-VL-Embedding, while keeping the remaining modules frozen. We use a learning rate of 5×10^{-5} and train for 6 epochs per stage.

Joint FT. Joint FT is implemented as a non-continual reference baseline. We merge the training data from all six stages and perform a single supervised fine-tuning run on the aggregated dataset. As in the sequential setting, we adopt LoRA fine-tuning and update only the LLM part of Qwen3-VL-Embedding. We use the same learning rate of 5×10^{-5} and train for 6 epochs.

Algorithm 1: Auto-K Multi-Prototype Dictionary Construction

Input: Train embeddings $\{(\mathbf{e}_i, y_i)\}_{i=1}^{N}$ with script-aware lexical entries y_i, $K_{\max}$=32, sample cap S=256

Output: Prototype bank $\mathbf{P}$ (flattened), pointer array π

Group embeddings by entry: $\mathbf{E}_y \leftarrow \{\mathbf{e}_i : y_i{=}y\}$.;

Initialize $\mathbf{P} \leftarrow [\,]$, $\pi \leftarrow [0]$;

foreach *lexical entry* y **do**

  $K_y^{\max} \leftarrow \min(K_{\max}, \lfloor\sqrt{|\mathbf{E}_y|}\rfloor)$;

  if $K_y^{\max} < 2$ **then**

    $K_y \leftarrow 1$;

  else

    Sample $\tilde{\mathbf{E}}_y \subseteq \mathbf{E}_y$ with $|\tilde{\mathbf{E}}_y| = \min(|\mathbf{E}_y|, S)$;

    $K_y \leftarrow \arg\max_{k\in[2,K_y^{\max}]} \mathrm{Silhouette}_{\cos}(\tilde{\mathbf{E}}_y, \mathrm{SphKMeans}(\tilde{\mathbf{E}}_y, k))$;

  $\mathbf{C}_y \leftarrow \mathrm{SphKMeans}(\mathbf{E}_y, K_y)$;

  Append $\mathbf{C}_y$ to $\mathbf{P}$; append new prefix sum to π;

ER. ER is implemented on top of the same LoRA fine-tuning setting as Seq FT. It maintains a replay buffer with a size of 10,000 and jointly trains on the current-stage data and replayed samples from previous stages.

LwF. LwF is implemented through stage-to-stage knowledge distillation, where the model from the previous stage provides soft targets to regularize the current-stage optimization. The same LoRA fine-tuning configuration is used.

EWC. EWC introduces parameter regularization based on the estimated importance of parameters learned from previous stages. It is implemented under the same LoRA fine-tuning setting as the other Qwen3-VL-Embedding baselines.

iCaRL. iCaRL maintains an exemplar buffer with a size of 10,000 and uses stored exemplars from previous stages to alleviate forgetting during continual adaptation. It follows the same LoRA fine-tuning configuration.

DER++. DER++ maintains a replay buffer with a size of 10,000 and extends standard rehearsal with additional distillation-based regularization on replayed samples. It follows the same LoRA fine-tuning configuration.

F Additional Experimental Results

This section provides additional experimental results omitted from the main paper due to space constraints. We report the full stage-wise ablation trajectories and provide detailed analysis of the diagnostic factors studied in the main paper.

F.1 Full Stage-wise Ablation Results

Table 4 reports the full stage-wise ablation results under the continual onboarding protocol and the zero-shot candidate retrieval track. The compact diagnostic table in the main paper reports only the final-stage metrics for readability, while this table shows the complete AA1–AA6 trajectory.

F.2 Detailed Ablation Analysis

Model scale. Comparing AMR-CCR (2B) and AMR-CCR (8B) shows a clear accuracy–stability trade-off. The 8B model achieves substantially higher final retrieval accuracy (58.6/83.1 vs. 50.4/72.9 at AA6), suggesting that larger pretrained vision–language embeddings better separate dense near-duplicate lexical entries. In contrast, the 2B model has lower forgetting (1.2/1.7 vs. 1.9/2.7), indicating that smaller models can be more stable under continual updates. The zero-shot results show different behavior at different ranks: the 2B model has higher ZS@1, while the 8B model has higher ZS@20, suggesting that top-rank calibration and candidate recall may benefit from different model properties.

Script-conditioned adaptation and routing. Removing both SIA and SAR causes a large drop from 58.6/83.1 to 28.7/55.3 at AA6 and increases forgetting from 1.9/2.7 to 9.5/9.7. This confirms that continual updates without script-conditioned modular calibration can severely break cross-stage

Table 4: Full ablation results under the continual protocol and the zero-shot (ZS) track. Continual results report Top-1/Top-10; ZS reports Top-1/Top-20. Gold denotes ground-truth routing, and RS denotes random sampling.

	Variant	AA1↑	AA2↑	AA3↑	AA4↑	AA5↑	AA6↑	FGT↓	ZS↑
MS	2B	57.9/85.0	39.5/66.0	37.6/63.6	46.1/70.6	46.9/70.4	50.4/72.9	1.2/1.7	20.9/38.6
Modules	w/o SIA & SAR	67.2/87.6	24.1/43.7	21.6/48.3	22.9/49.1	16.5/38.1	28.7/55.3	9.5/9.7	–/–
	w/o SAR (Gold)	68.9/90.3	56.2/85.7	52.4/83.4	57.5/85.8	59.3/85.3	61.2/86.8	0.9/1.7	15.9/48.8
BufS	5k	68.2/80.8	55.1/84.2	49.2/79.9	54.3/81.5	55.1/82.8	57.2/80.8	2.8/3.6	14.3/43.9
	15k	68.4/90.1	55.7/84.9	50.7/80.4	56.3/83.9	57.9/82.2	59.1/83.8	1.7/2.4	15.2/47.0
PC	Mean	67.6/87.9	41.0/58.1	40.9/61.9	48.8/69.0	47.1/68.1	48.5/70.2	1.6/2.2	–/–
	RS (M=8)	68.2/88.0	42.2/59.1	40.6/61.8	48.5/69.0	45.1/63.2	46.4/65.4	1.6/2.1	–/–
Data-S1	Image-only	20.7/46.5	16.9/40.0	25.7/49.7	35.5/59.5	37.1/61.1	43.4/65.8	1.1/1.9	9.9/25.9
	AMR-CCR (8B)	68.4/90.0	55.5/84.6	50.7/80.3	56.1/83.6	57.1/81.8	58.6/83.1	1.9/2.7	15.2/46.8

compatibility. The gold-routing variant, which uses the ground-truth script to select the adapter, reaches 61.2/86.8 at AA6 and 15.9/48.8 on zero-shot retrieval. This provides an oracle upper bound for SAR and shows that the learned router is close to oracle routing in this setting.

Replay buffer size. The buffer-size comparison shows that replay helps maintain cross-stage compatibility. Reducing the buffer to 5k decreases AA6 to 57.2/80.8 and increases forgetting to 2.8/3.6, while increasing it to 15k improves AA6 to 59.1/83.8 and reduces forgetting to 1.7/2.4. The improvement from the default 10k buffer to 15k is modest, suggesting that 10k provides a practical trade-off between memory cost and retrieval performance.

Prototype construction. Replacing the Auto-K multi-prototype dictionary with a single mean prototype or random sampling substantially degrades final retrieval accuracy, reducing AA6 to 48.5/70.2 and 46.4/65.4, respectively. In contrast, forgetting remains similar across these variants. This indicates that the main benefit of multi-prototype dictionary construction is not simply reducing forgetting, but better covering diverse writing styles and acquisition modes within the same lexical entry.

Textual supervision. The image-only variant removes meaning and shape descriptions from Phase-I training. Its performance drops to 43.4/65.8 at AA6 and 9.9/25.9 on zero-shot retrieval, compared with 58.6/83.1 and 15.2/46.8 for AMR-CCR (8B). This shows that language-grounded supervision provides useful information beyond image-only contrastive learning, especially for image-to-text candidate retrieval in the zero-shot track. In this paper, textual supervision consists of character-level meaning descriptions and image-level shape descriptions.

G Qualitative Retrieval Examples and Failure Cases

Figure 10 compares Top-3 retrieval results of the frozen pretrained embedding model and AMR-CCR on representative query samples. Overall, AMR-CCR more often retrieves script-consistent and structurally plausible candidates. In several cases, the frozen pretrained model is distracted by coarse visual similarity and returns candidates from an incorrect script or a less relevant structural neighborhood, whereas AMR-CCR retrieves the exact target at Top-1.

Even when the exact target is not ranked first, AMR-CCR tends to produce more useful candidate lists. Remaining errors mainly occur for highly stylized or visually confusable glyphs, suggesting that fine-grained glyph discrimination remains challenging even with language-grounded retrieval.

H Release and Ethical Considerations

We plan to release EvoCON split files, language-grounded annotations, evaluation scripts, and the AMR-CCR reference implementation for reproducible research. The released benchmark is intended for cultural-heritage digitization and digital humanities research. EvoCON is derived from existing glyph resources and does not contain personal or sensitive human-subject data.

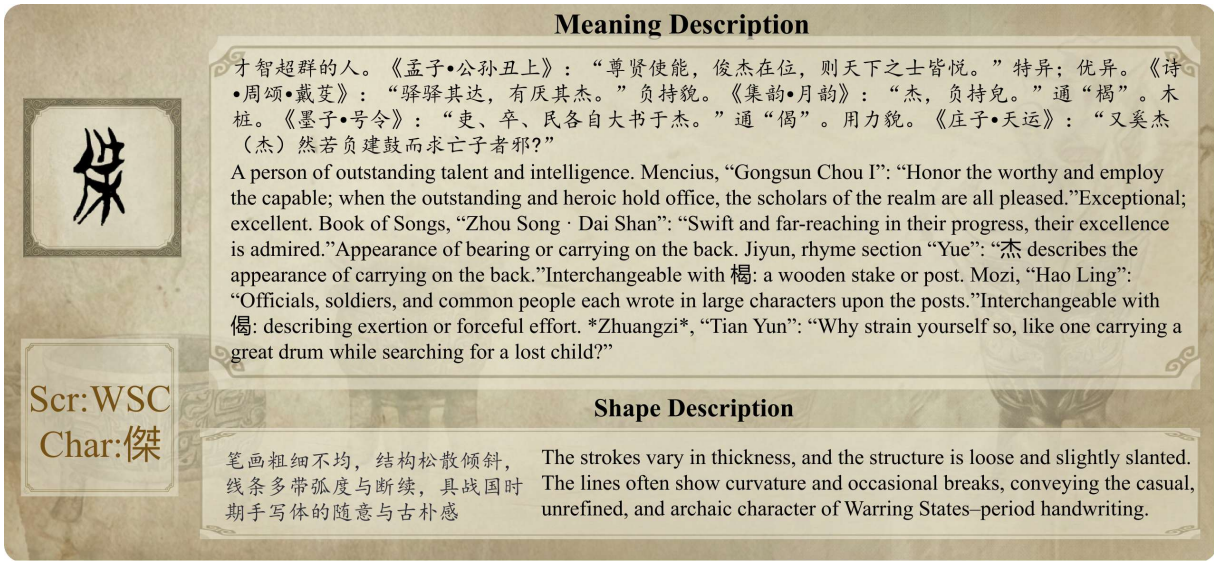


Figure 5: Representative EvoCON example (1/5), showing the glyph image, script identity, modern-character identity, meaning description, and shape description.

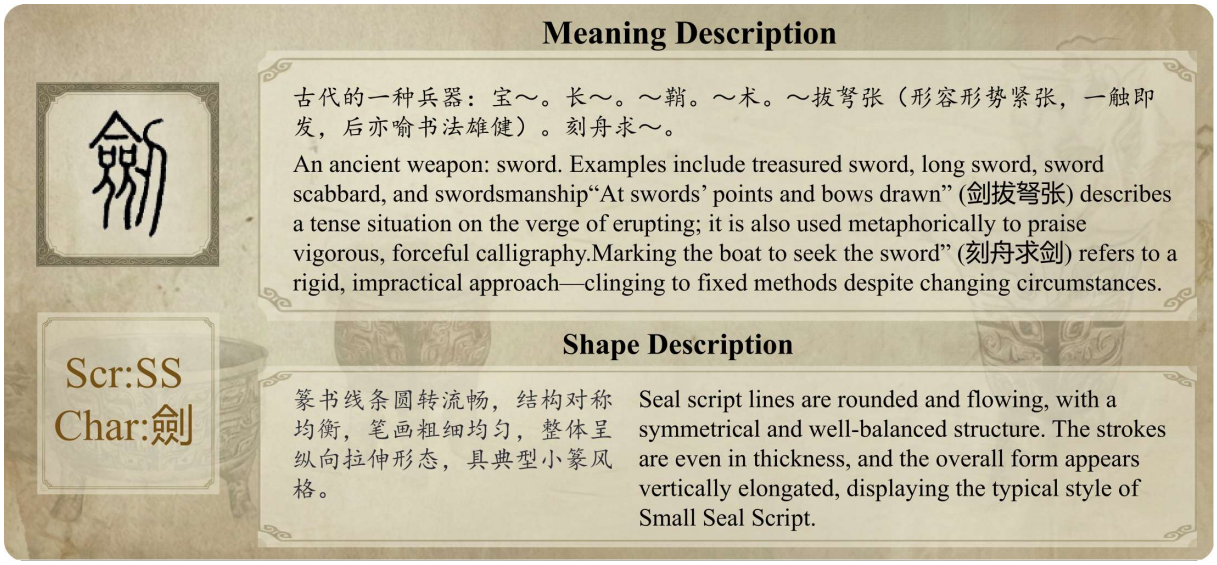


Figure 6: Representative EvoCON example (2/5).

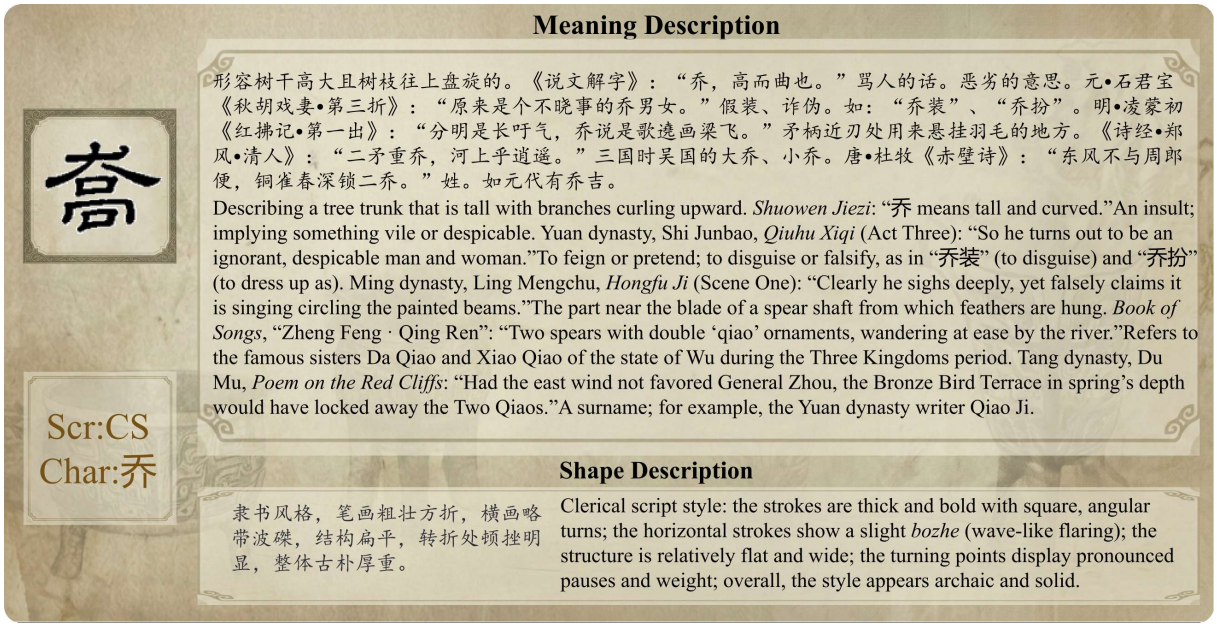


Figure 7: Representative EvoCON example (3/5).

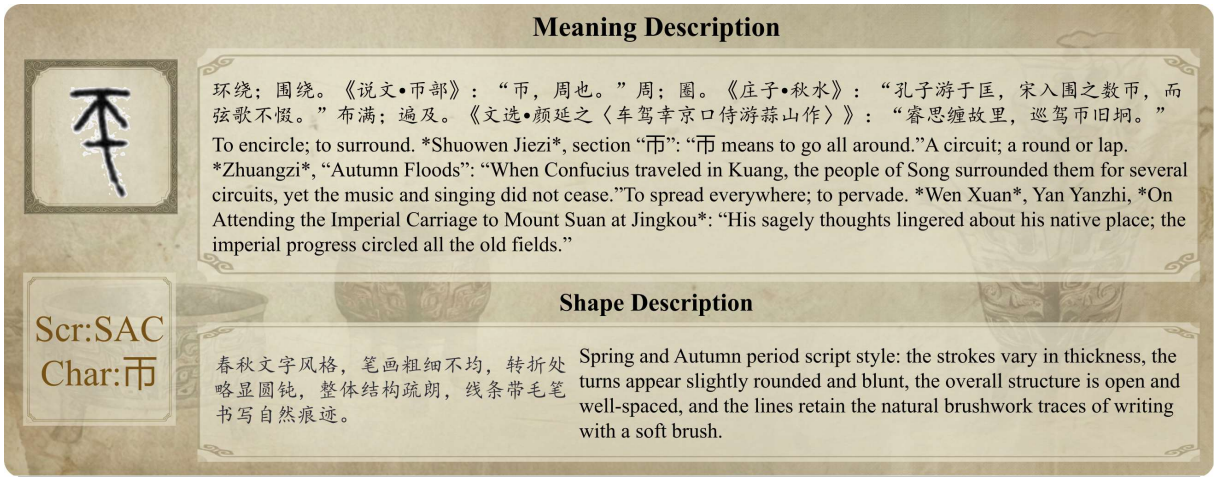


Figure 8: Representative EvoCON example (4/5).

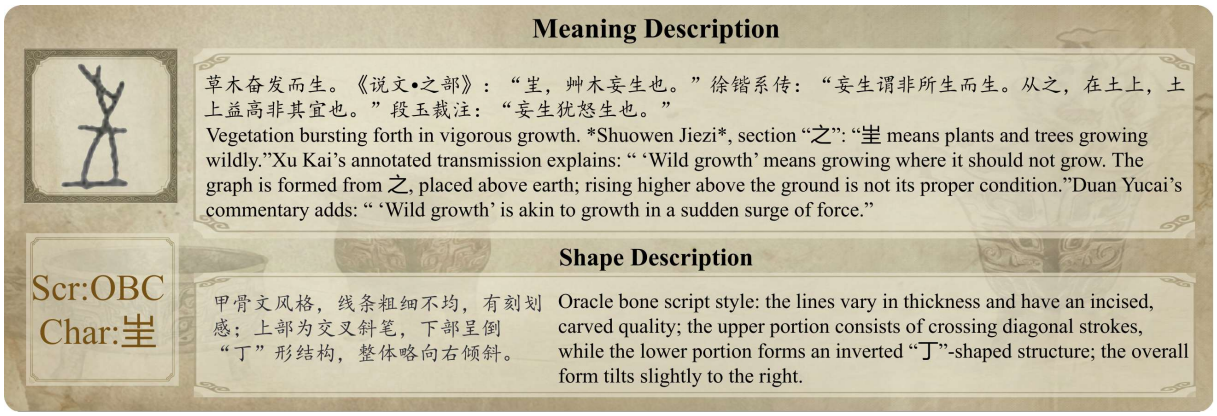


Figure 9: Representative EvoCON example (5/5).

Query Image	GT	Pretrained (Top3)	Ours (Top3)
	(丙,SAC)	Top1:(句,WSC) Top2:(君,WSC) Top3:(牙,WSC)	Top1:(丙,SAC) Top2:(曰,SAC) Top3:(哀,SAC)
	(滿,WSC)	Top1:(塙,WSC) Top2:(齊,WSC) Top3:(兩,WSC)	Top1:(滿,WSC) Top2:(雨,WSC) Top3:(兩,WSC)
	(三,SS)	Top1:(次,SS) Top2:(辵,CS) Top3:(差,SS)	Top1:(式,SS) Top2:(哉,SS) Top3:(三,SS)
	(丘,WSC)	Top1:(肱,BI) Top2:(顯,BI) Top3:(養,BI)	Top1:(丘,WSC) Top2:(公,WSC) Top3:(敘,WSC)
	(㠯,BI)	Top1:(申,BI) Top2:(德,BI) Top3:(九,BI)	Top1:(㠯,BI) Top2:(臣,BI) Top3:(右,BI)
	(妈,OBC)	Top1:(婦,OBC) Top2:(娶,OBC) Top3:(娠,OBC)	Top1:(妈,OBC) Top2:(娘,OBC) Top3:(姼,OBC)

Figure 10: Qualitative retrieval cases comparing the frozen pretrained embedding model and AMR-CCR. For each query image, we show the ground-truth lexical entry and the Top-3 retrieved results.