

Preference-Aligned Data Synthesis for Fine-Grained Intent Detection

Sitong Shen

Dalian University of Technology
shensitong.dut@foxmail.com

Xiaotong Zhang*

Dalian University of Technology
zhangxt@hotmail.com

Abstract

Fine-grained intent detection becomes brittle as label spaces expand with many semantically adjacent intents. Although large language models (LLMs) can synthesize labeled utterances to reduce annotation cost, the resulting pseudo data are often ambiguous across intents and repetitive within intents. Existing synthesis pipelines mainly rely on prompt engineering and filtering, without updating generator parameters. We propose a preference-aligned synthesis framework that automatically mines supervision to optimize the generator. Each intent is represented by multi-centroid prototypes, and block-conditioned generation supports diversity-aware candidate construction. From these candidates, we build preferred blocks that are both prototype-consistent and diverse, and rejected blocks composed of confusable negatives from neighboring intents. These preference pairs are used to optimize the generator with Direct Preference Optimization (DPO), together with iterative refresh to mine harder confusions over time. Experiments show consistent gains for downstream intent classification in zero-shot and few-shot settings, while remaining competitive in fully supervised settings. We further observe improved generator reliability and strong transfer across multiple backbone LLM families.

1 Introduction

Intent detection aims to infer a user’s goal from an utterance by assigning it to an intent label, and remains a core component of spoken and text-based natural language understanding systems (Coucke et al., 2018; Weld et al., 2022). Its importance is further increasing in LLM-based agents, where intent understanding supports action selection, planning, and tool use across multi-turn interactions (Li, 2025; Qian et al., 2024; Tang et al., 2025; Kim

et al., 2024). In practice, however, intent classification rarely operates in a fixed label space: intent inventories continually expand and become more fine-grained, while collecting labeled data for emerging intents remains costly. This makes generalized zero-shot and few-shot intent detection increasingly important (Zhang et al., 2021).

To alleviate data scarcity, recent work has explored Large Language Models (LLMs) as generators of pseudo-labeled training samples for downstream classifiers (Ye et al., 2022). Yet fine-grained intent synthesis remains difficult for two reasons. First, **inter-class semantic overlap** makes closely related intents hard to separate, so LLMs may miss subtle distinctions and generate ambiguous utterances that blur decision boundaries. Second, **intra-class homogeneity** causes LLMs to overproduce high-probability generic patterns, resulting in repetitive and lexically narrow samples that limit downstream generalization. Existing remedies mainly rely on prompt engineering or iterative prompting (Zhang et al., 2025). While useful, such methods steer generation only at inference time and do not directly update the generator parameters, so their improvements are often limited or unstable. Moreover, classifier accuracy alone does not fully reveal whether the generator itself produces reliable synthetic data, especially in true zero-shot settings where labeled validation data are unavailable. This calls for a synthesis framework that improves the generator at the parameter level and a generator-level signal that can monitor synthesis quality during iterative refinement.

To address this problem, we move from *prompt-only steering* to *preference-aligned synthesis* with automatically mined supervision with automatically mined supervision. Our framework maintains an intent-specific anchor set and derives multiple prototypes to capture diverse valid realizations of the same intent, rather than relying on a single semantic center. The generator then produces ut-

*Corresponding author.

terance blocks, from which we mine *prototype-consistent and non-redundant* preferred samples as well as *confusable hard negatives* from neighboring intents. These preferred/rejected pairs are used to optimize the generator with Direct Preference Optimization (DPO) (Rafailov et al., 2023), allowing the model to directly contrast diverse in-class positives against boundary-challenging negatives. We further introduce iterative preference refresh, which periodically regenerates candidates and re-mines preference pairs as the generator improves. In addition, we use generator-level evaluation to monitor synthesis reliability and determine when further refresh rounds cease to be beneficial.

Experiments on three benchmarks show that the proposed framework consistently improves intent classification under zero-shot, few-shot, and full-shot settings over strong prompting and augmentation baselines. The gains are accompanied by better generator reliability, and they transfer across multiple backbone LLM families, suggesting that the benefit does not depend on a single generator architecture.

Our main contributions are as follows:

- We propose a preference-aligned synthesis framework that updates the generator with automatically mined preference supervision for fine-grained intent synthesis.
- We design a diversity-aware positive mining and confusability-aware negative mining strategy, and optimize the generator with DPO and iterative preference refresh.
- Experiments on three benchmarks show consistent gains in zero-/few-shot intent classification, improved generator reliability, and transferability across multiple LLM backbones.

2 Related Work

2.1 Low-Resource Fine-Grained Intent Detection

Low-resource intent detection focuses on zero- and few-shot intent recognition with scarce labeled data. Early studies mainly improved discriminative models through label semantics, natural language inference (NLI)-style reformulation (Chen et al., 2016; Zhang et al., 2020), or episodic meta-learning (Finn et al., 2017; Snell et al., 2017). Recent work increasingly adopts instruction- and generation-based formulations, including hybrid systems that

combine small task models with LLMs for improved low-resource performance (Zhang et al., 2024; Chen and Wang, 2025). For fine-grained intent spaces with high semantic overlap, differential prompting and contrastive feedback generation reduce near-miss confusions by modeling inter-class relations (Zhang et al., 2025). In contrast, we align an LLM data generator via automatically mined preferences to produce intent-faithful yet diverse utterances, complementing these approaches.

2.2 LLM-Based Data Synthesis for Intent Classification

LLMs have been widely used to synthesize pseudo-labeled data for training intent classifiers under limited supervision (Yoo et al., 2021; Ye et al., 2022). Most approaches keep the generator frozen and rely on prompt engineering with heuristic filtering or iterative refinement (Chen and Shu, 2023; Madaan et al., 2023). More recent work improves synthetic data through multi-stage generation, parameter-efficient fine-tuning (PEFT), or diversity-oriented augmentation (Lin et al., 2024; Patwa et al., 2024; Wang et al., 2025). For fine-grained intents, differential generation with contrastive feedback explicitly mitigates boundary errors (Zhang et al., 2025). Our method instead performs parameter-level preference alignment using prototype-aligned positives and weighted hard negatives to address both inter-class overlap and intra-class homogeneity.

2.3 Preference Optimization for LLM Alignment

Preference optimization aligns LLMs from comparative feedback, popularized by reinforcement learning from human feedback (RLHF) pipelines (Christiano et al., 2017; Ouyang et al., 2022). Direct Preference Optimization (DPO) learns directly from preference pairs with a classification-style objective, avoiding explicit reward modeling and on-policy reinforcement learning (RL) (Rafailov et al., 2023); subsequent variants adjust preference modeling assumptions or reduce complexity (Hong et al., 2024; Amini et al., 2024), and preference data can be scaled via AI feedback (Bai et al., 2022; Lee et al., 2023). While most work targets general assistant behaviors, we use preference optimization to align a *task-specific generator* for fine-grained intent data synthesis by automatically mining structured positive/negative blocks that encode boundary-safe yet diverse realizations.

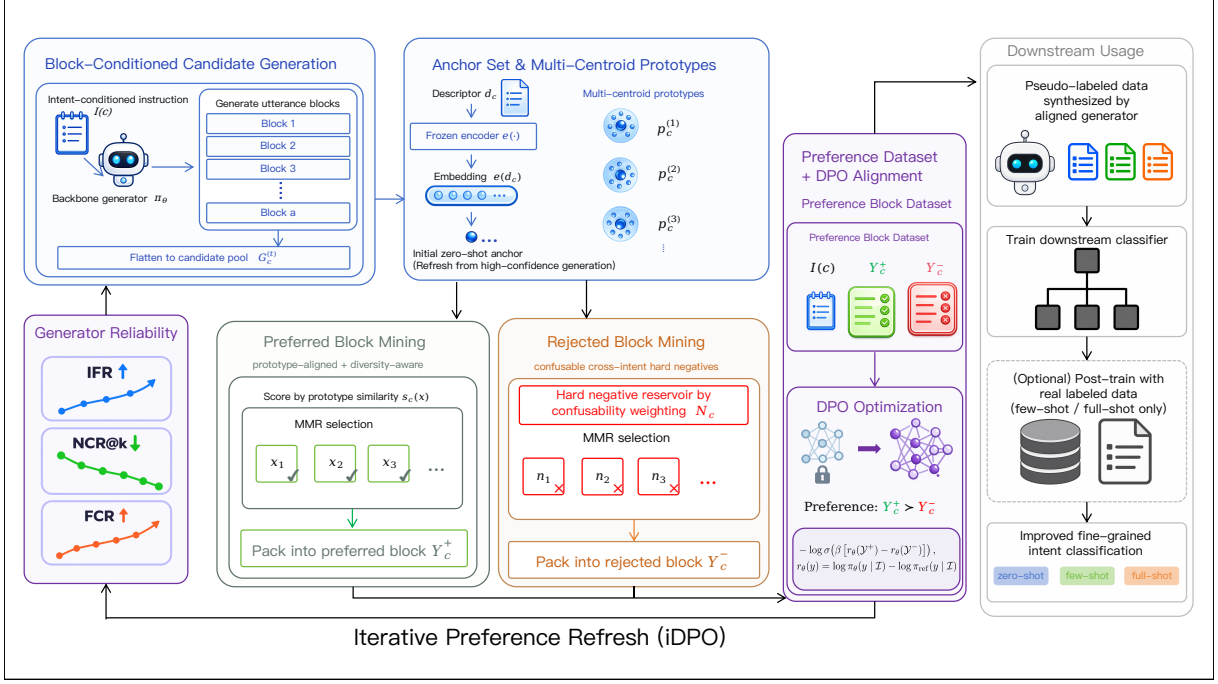


Figure 1: Overview of our preference-aligned synthesis framework. At each refresh round, the current generator produces intent-conditioned candidate blocks. Intent prototypes are used to construct diverse preferred blocks and cross-intent confusable rejected blocks, which form preference pairs for DPO. The updated generator then starts the next refresh round. After generator-side reliability stops improving, the aligned generator synthesizes pseudo-labeled data for downstream classifier training. Optional real examples are used only to post-train the classifier and never enter generator alignment.

3 Methodology

3.1 Problem Formulation

Let $\mathcal{X}$ denote the space of user utterances and $\mathcal{C}$ the set of intent labels. Given an utterance $x \in \mathcal{X}$, the task is to predict its intent $y \in \mathcal{C}$. At test time we observe unlabeled utterances $\mathcal{D}_{\text{test}} = \{x_j\}_{j=1}^m$. Optional labeled supervision is denoted as $\mathcal{D}_t = \{(x_i, y_i)\}_{i=1}^n$, where $n \ll m$ (few-shot) or $n = 0$ (zero-shot). Each intent $c \in \mathcal{C}$ is associated with a natural-language descriptor d_c .

3.2 Method Overview

Our method separates generator alignment from downstream classifier training (Figure 1). For each intent c , an *anchor set* contains its descriptor and reliable generated utterances; clustering their embeddings gives *multi-centroid prototypes* that represent different valid realizations of the intent. At refresh round r , the current generator $\pi_\theta^{(r)}$ produces ℓ -utterance *candidate blocks*. Prototype-aligned and non-redundant candidates are repacked as *preferred blocks*, whereas utterances generated for other intents but highly similar to c form *rejected blocks*. Pairing the same intent instruction with a preferred and a rejected block yields a DPO preference tuple.

The updated generator then regenerates candidates and re-mines preferences. After generator-side reliability stops improving, the final generator synthesizes pseudo-labeled data for classifier f ; optional real examples are used only to post-train f .

3.3 Anchor Set

To formalize intent alignment, we construct an *Anchor Set* for each intent c . While anchors originate as text (e.g., utterances or short descriptors) in a text-level set $\tilde{\mathcal{A}}_c$, all subsequent operations are performed in the embedding space. Specifically, a frozen encoder $e(\cdot) : \text{Text} \rightarrow \mathbb{R}^d$ maps these textual anchors to form the embedding-level anchor set:

$$\mathcal{A}_c = \{e(z) \mid z \in \tilde{\mathcal{A}}_c\} \subset \mathbb{R}^d. \quad (1)$$

Zero-Shot Anchor Initialization and Refresh. We initialize anchors from label semantics, where d_c is the label’s textual description used as anchor text:

$$\tilde{\mathcal{A}}_c^{(0)} = \{d_c\}, \quad \mu_c^{(0)} = e(d_c), \quad \mathcal{A}_c^{(0)} = \{\mu_c^{(0)}\}. \quad (2)$$

In later refresh rounds $r \geq 1$, we update the anchor to the semantic center of newly generated high-

confidence texts:

$$\mu_c^{(r+1)} = \text{norm} \left(\frac{1}{|\mathcal{H}_c^{(r)}|} \sum_{x \in \mathcal{H}_c^{(r)}} e(x) \right). \quad (3)$$

where $\mathcal{H}_c^{(r)} \subset \mathcal{G}_c^{(r)}$ is a high-confidence subset. Here $\text{norm}(\cdot)$ denotes standard ℓ_2 -normalization.

Optionally, we append these texts to the anchor set:

$$\tilde{\mathcal{A}}_c^{(r+1)} \leftarrow \tilde{\mathcal{A}}_c^{(r)} \cup \mathcal{H}_c^{(r)}. \quad (4)$$

Bootstrapping Drift and Mitigation. In the zero-shot regime, building $\mathcal{H}_c^{(r)}$ from model generations may cause *bootstrapping drift*, where spurious but frequent patterns bias $\mu_c^{(r)}$. We mitigate drift by (i) mining cross-intent hard negatives (Sec. 3.6) to form rejected blocks for DPO (Sec. 3.7), suppressing boundary-confusable modes under the updated policy; and (ii) applying diversity control based on Maximal Marginal Relevance (MMR) during anchor pruning and block construction so that no single mode dominates the bootstrap signal. The formal definition of the MMR operator is given in Appendix D.3.

Multi-Centroid Prototypes. When we maintain multiple anchors per class, we cluster $\mathcal{A}_c$ into $k \geq 1$ groups and obtain normalized centroids $\mathbf{P}_c \in \mathbb{R}^{k \times d}$. For any utterance x , its prototype similarity to intent c is

$$s_c(x) = \max_{j \in \{1, \dots, k\}} \langle e(x), \mathbf{P}_c^{(j)} \rangle. \quad (5)$$

We denote the mean prototype vector by

$$\bar{\mathbf{p}}_c = \text{norm} \left(\frac{1}{k} \sum_{j=1}^k \mathbf{P}_c^{(j)} \right). \quad (6)$$

If the anchor set $\tilde{\mathcal{A}}_c$ grows large, we prune anchors with an MMR-based diversity filter in embedding space; the full selection rule is deferred to Appendix D.3.

3.4 Block-Conditioned Candidate Generation

Instead of generating single utterances independently, we adopt a *Block-Conditioned Generation* strategy where an ℓ -utterance block is treated as the atomic unit for both generation and optimization.

Let g_{base} denote the backbone generator and $\mathcal{I}(c)$ an intent-conditioned instruction. The model generates a block

$$\mathbf{U}_c = [x_1, \dots, x_\ell] \sim p_{g_{\text{base}}}(\cdot \mid \mathcal{I}(c), \text{Constraints}) \quad (7)$$

where the constraints enforce (i) intent-label consistency, (ii) intra-block lexical/stylistic diversity, and (iii) exclusion of overlapping sibling intents or meta-data leakage.

Candidate Pool Construction. At refresh round r , by repeating block generation a times, we obtain a set of generated blocks $\{\mathbf{U}_c^{(t,r)}\}_{t=1}^a$. For mining and retrieval, we use their flattened utterance pool

$$\mathcal{G}_c^{(r)} = \bigcup_{t=1}^a \mathbf{U}_c^{(t,r)} = \{x_c^{(r,j)}\}_{j=1}^{a \cdot \ell}, \quad (8)$$

where $K = a \cdot \ell$. Any training instance used by DPO is always packed back into an ℓ -line block.

3.5 Prototype-Aligned Positives: Diversity-Aware Preferred Blocks

From the candidate pool $\mathcal{G}_c^{(r)}$, we mine multiple preferred blocks $\{\mathcal{Y}_{c,b}^+\}_{b=1}^M$ for each intent c . Constructing M preferred blocks requires $M\ell$ positive utterance slots, which are selected without replacement from $\mathcal{G}_c^{(r)}$. Although blocks are atomic during generation/optimization, we may assemble a preferred block by selecting ℓ utterances from the pooled candidates and then packing them into an ℓ -line block, which makes the preference signal less sensitive to any single generated block’s local artifacts.

Simply taking top- ℓ by $s_c(x)$ can be redundant, so we use MMR with prototype similarity as relevance; the generic operator is defined in Appendix D.3. Let $\Omega_{c,1} = \mathcal{G}_c^{(r)}$. For $b = 1, \dots, M$, we define

$$\begin{aligned} \phi_+(x) &= s_c(x), & S_{c,b} &= \text{MMR}(\Omega_{c,b}, \phi_+, \ell), \\ \mathcal{Y}_{c,b}^+ &= \text{Pack}(S_{c,b}), & \Omega_{c,b+1} &\leftarrow \Omega_{c,b} \setminus S_{c,b}. \end{aligned} \quad (9)$$

This sequential packing yields diverse prototype-aligned positives within the same intent.

3.6 Confusability-Weighted Hard Negative Reservoir

To sharpen fine-grained boundaries, we construct a *Confusability-weighted Hard Negative Reservoir* $\mathcal{N}_c$, which serves as a cross-intent pool of highly confusable negatives for forming informative rejected blocks. We first define an explicit confusability weight from intent c' to c based on prototype proximity:

$$w(c' \rightarrow c) = \frac{\exp(\gamma \langle \bar{\mathbf{p}}_{c'}, \bar{\mathbf{p}}_c \rangle)}{\sum_{\tilde{c} \neq c} \exp(\gamma \langle \bar{\mathbf{p}}_{\tilde{c}}, \bar{\mathbf{p}}_c \rangle)}, \quad c' \neq c, \quad (10)$$

where $\gamma > 0$ controls the sharpness. Larger $w(c' \rightarrow c)$ indicates that c' is more confusable with c .

Reservoir Construction for Retrieving Cross-Intent Confusable Negatives. For each source intent $c' \neq c$, we first retrieve a top- Q shortlist from $\mathcal{G}_c^{(r)}$ according to the target-intent similarity $s_c(\cdot)$. Here, Q controls only the per-source retrieval depth. To construct M rejected blocks, we allocate a total budget of $M\ell$ negative utterances across the source intents proportionally to $w(c' \rightarrow c)$, using integer rounding and remainder redistribution. We then merge the allocated candidates into $\mathcal{N}_c^{(r)}$. Full allocation details are provided in Appendix B.3.

Rejected Block Sampling via MMR Over Hard-Negative Pool. Let $\mathcal{N}_c$ denote the hard-negative reservoir for intent c , and initialize $\mathcal{N}_{c,1} = \mathcal{N}_c$, where b indexes rejected blocks packed sequentially without replacement. We construct rejected blocks $\{\mathcal{Y}_{c,b}^-\}_{b=1}^M$ by selecting ℓ utterances from the hard-negative pool and then packing them into an ℓ -line block:

$$\begin{aligned} \phi_-(x) &= \langle e(x), \bar{\mathbf{p}}_c \rangle, \\ \mathcal{Y}_{c,b}^- &= \text{Pack}(\text{MMR}(\mathcal{N}_{c,b}, \phi_-, \ell)), \\ \mathcal{N}_{c,b+1} &\leftarrow \mathcal{N}_{c,b} \setminus \text{Unpack}(\mathcal{Y}_{c,b}^-). \end{aligned} \quad (11)$$

3.7 Preference Block Dataset and DPO Objective

We construct the preference dataset $\mathcal{D}_{\text{pref}}$ by pairing the prompt with mined positive/negative blocks:

$$(\mathcal{I}(c), \mathcal{Y}_{c,b}^+, \mathcal{Y}_{c,b}^-) \in \mathcal{D}_{\text{pref}}, \quad b = 1, \dots, M,$$

where both $\mathcal{Y}^+$ and $\mathcal{Y}^-$ are formatted as ℓ -line blocks, consistent with Sec. 3.4. Thus, each intent contributes M block-level preference tuples per refresh round, requiring $M\ell$ preferred and $M\ell$ rejected utterance slots.

We optimize the generator using Direct Preference Optimization (DPO):

$$\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}_{\xi \sim \mathcal{D}_{\text{pref}}} \left[\log \sigma(\beta(r_\theta(\mathcal{Y}^+) - r_\theta(\mathcal{Y}^-))) \right], \quad (12)$$

where $r_\theta(y) = \log \pi_\theta(y | \mathcal{I}) - \log \pi_{\text{ref}}(y | \mathcal{I})$, π_{ref} is a frozen reference policy and β controls deviation from π_{ref} .

3.8 Iterative Preference Refresh

To progressively improve the generator, we adopt an iterative training strategy over R rounds. At each round, the current policy regenerates candidate pools, updates anchor prototypes using high-confidence samples, re-mines preference blocks, and performs one round of DPO optimization.

As the generator improves, the mined negatives become increasingly challenging, forming a self-reinforcing curriculum. Consequently, this dynamic updating mechanism compels the model to learn more robust and discriminative intent representations. The full procedure is summarized in Figure 1.

3.9 Generator Reliability Evaluation

Beyond downstream accuracy, we evaluate generator reliability—and guide the number of iterative refresh rounds—using three metrics (formally defined in Appendix D.1): *Intent Fidelity Rate* (IFR), *Neighbor Confusion Rate at k* (NCR@ k), and *Format Compliance Rate* (FCR). Evaluations strictly rely on an independent embedding-based verifier $h(\cdot)$ (Appendix D.2), which assigns utterances to the most similar intent descriptor. To prevent data leakage, $h(\cdot)$ is strictly isolated from generation, preference mining, and downstream training. Specifically, IFR measures successful assignment to the target intent (higher is better), while NCR@ k quantifies misclassification into the k most confusable neighbor intents ranked by prototype similarity (lower is better).

Format Compliance Rate. FCR measures the percentage of generated blocks that follow the requested ℓ -utterance format. A block is counted as compliant if it contains exactly ℓ utterances and does not include empty lines, label leakage, or extra explanations. The full rule set is given in Appendix C.4. In practice, we increase the refresh round only while these generator-side signals improve, and stop when they saturate or begin to degrade.

3.10 Post-Training the Downstream Classifier with Real Data

When labeled data $\mathcal{D}_t$ is available (few-shot or fully supervised settings), we do not inject it into the generator alignment or anchor construction. Instead, we first train the downstream intent classifier f on the synthesized pseudo-labeled dataset $\tilde{\mathcal{D}}$, and then *post-train* f on $\mathcal{D}_t$ using standard supervised learn-

Dataset	Classes	Average Length	Train/Valid/Test Samples
BANKING77 (Casanueva et al., 2020)	77	11.8	8622 / 1540 / 3080
CLINC150 (Larson et al., 2019)	150	8.3	15000 / 3000 / 4500
HWU64 (Liu et al., 2021)	64	6.6	8954 / 1076 / 1076

Table 1: Statistics of the intent classification datasets used in our experiments.

ing. This strategy leverages synthetic data for broad coverage while using real labeled examples to calibrate decision boundaries and improve in-domain fidelity.

4 Experiment

4.1 Datasets

We evaluate on three intent classification benchmarks following prior protocol (Lin et al., 2023): BANKING77 (Casanueva et al., 2020), HWU64 (Liu et al., 2021), and CLINC150 (Larson et al., 2019) (see Table 1 for statistics).

For few-shot setting, we randomly sample k training instances per intent from the original training split. For zero-shot setting, no labeled data are used.

4.2 Baselines and Evaluation Metrics

We compare against strong prior methods in three groups: (i) low-resource intent classification models, (ii) traditional text augmentation, and (iii) recent LLM-based generation/augmentation approaches. The complete baseline list and their configurations are reported in Appendix A. Following (Zhang et al., 2025), we report classification accuracy under zero-shot, few-shot, and full-shot settings.

4.3 Implementation Details

Generator. We use **Qwen3-8B** (Yang et al., 2025) as the main backbone and apply **Low-Rank Adaptation (LoRA)** (Hu et al., 2022) for parameter-efficient updates. To examine whether the proposed framework generalizes beyond a single model family, we additionally conduct experiments with **Llama3-8B** (Team, 2024) and **Gemma2-9B** (Team et al., 2024) using the same training pipeline. We run R refresh rounds. For each round, we generate $B=500$ utterances per intent and select $N=100$ per intent from the final pool for downstream training.

Downstream Classifier. Following (Lin et al., 2023), we fine-tune **RoBERTa-base** (Liu et al.,

2019) under the standard protocol; additional hyperparameters are deferred to Appendix B.

5 Result Analysis

5.1 Main Results

We evaluate intent detection on BANKING77, HWU64, and CLINC150 under zero-shot and few-shot supervision (Table 2). Baselines are taken from (Lin et al., 2023) and (Zhang et al., 2025); best results are boldfaced. Full-shot results are reported in Appendix C.1. Detailed cross-backbone results are reported in Appendix C.2 (Table 6). The results demonstrate that our approach is effective across datasets with diverse label spaces and data distributions, and generalizes well across different backbones.

Zero-Shot Performance. In the label-only (zero-shot) setting, our method achieves state-of-the-art performance among label-only approaches, with the largest gain on **CLINC150**. Zero-shot accuracy is largely driven by the *semantic alignment* between label names and real utterances. Although diversity-aware selection and confusability-weighted hard negatives improve intra-class coverage and inter-class separation, a domain gap persists when label names underspecify intent scope. This is more pronounced in BANKING77, where labels such as `order_physical_card` omit pragmatic constraints, while CLINC150 labels are more self-contained and thus transfer more effectively to decision boundaries.

Few-Shot and Full-Shot Performance. In 5/10-shot regimes, our method ranks first: few-shot labels are *not* used to adapt the generator or anchor set, so gains mainly come from higher-quality and more diverse synthetic pretraining. In full-shot, improvements persist but diminish (Appendix C.1), as abundant supervision reduces reliance on synthetic data.

Methods	BANKING77			HWU64			CLINC150		
	0-shot	5-shot	10-shot	0-shot	5-shot	10-shot	0-shot	5-shot	10-shot
<i>(1) Specialized Low-Resource Intent Classification Models</i>									
RoBERTa-Base	33.41	74.04	84.27	45.86	75.56	82.90	27.04	87.99	91.55
RoBERTa-Large	35.33	78.99	86.08	47.43	74.44	84.11	29.78	89.89	93.56
USE	42.79	76.29	84.23	46.43	77.79	83.75	44.51	87.82	90.85
CONVERT	37.23	75.32	83.32	43.95	76.95	82.65	43.53	89.22	92.62
USE+CONVERT	45.22	77.75	85.19	49.04	80.01	85.83	51.98	90.49	93.26
CONVBERT	30.64	74.27	83.63	44.67	77.70	83.77	49.25	90.07	92.10
CONVBERT+MLM	30.81	74.42	83.99	43.83	78.81	84.52	49.73	90.16	92.75
DNNC	48.15	80.40	86.71	54.73	80.46	84.72	52.44	91.02	93.76
CPFT	48.63	80.86	87.20	55.41	82.03	87.13	53.11	92.34	94.18
<i>(2) Traditional Text Augmentation Methods</i>									
BackTrans	37.36	76.14	85.67	51.39	76.93	83.71	41.30	88.65	91.79
EDA	37.15	73.02	84.42	53.82	76.83	84.48	31.23	87.29	92.21
AEDA	41.77	76.89	85.54	54.92	78.28	84.30	48.64	88.28	91.85
AMR-DA	40.41	74.46	84.85	54.66	78.40	84.43	49.14	88.58	91.35
SSMBA	30.38	72.36	83.86	42.74	74.97	83.04	25.63	86.87	91.34
GENIUS	37.17	75.01	84.03	55.34	77.60	84.45	47.59	88.31	91.73
<i>(3) LLM-Based Generation/Augmentation Methods</i>									
ZeroGen	48.41	74.52	84.81	47.84	77.69	84.76	53.36	88.46	91.56
CoDa	58.12	79.72	85.98	59.34	78.69	85.02	66.08	90.41	92.08
PromptMix	75.37	81.43	86.13	74.55	81.91	85.20	74.27	91.68	92.10
ICDA-L	–	83.90	89.12	–	81.97	86.94	–	92.41	94.73
IDG-CF	75.26	84.41	89.67	75.05	82.54	87.51	75.63	93.20	94.75
Ours	78.66	84.92	90.39	80.48	86.25	88.75	83.64	94.18	95.36

Table 2: Intent detection accuracy (%) in zero-shot and few-shot settings on three datasets. Baselines are grouped into specialized low-resource intent classification models, traditional text augmentation methods, and LLM-based generation/augmentation methods. **Ours** reports results with **Qwen3-8B** as the generator backbone. Results for other generator backbones are deferred to Appendix C.2.

5.2 Generalization Across Generator Backbones

As shown in Table 6, results with **Llama3-8B** and **Gemma2-9B** exhibit the same trend as **Qwen3-8B**, confirming that our method generalizes well across different generator backbones.

5.3 Ablation Study

To validate the contribution of individual components, we conduct ablation studies on BANKING77 and CLINC150 under the zero-shot and 5-shot settings. We take our full framework as the reference system and *disable one component at a time* while keeping all other settings unchanged. Results are reported in Table 3. **Unless otherwise stated, all variants are trained with the same iterative refresh budget ($R=2$) for controlled com-**

parison.

Table 3 shows that every component contributes positively. In particular, **iterative refresh** and **DPO** consistently improve performance, showing the value of refreshed preferences and explicit positive-negative contrasts. Hard-negative design further improves performance: boundary-focused negatives and **confusability-weighted budgeting** outperform random or uniform sampling, especially on harder intent boundaries. Finally, removing **MMR** causes the largest drop, highlighting that **positive-sample diversity is crucial**. Naive top- ℓ selection tends to over-emphasize near-duplicate patterns, encouraging the model to overfit to a few linguistic modes and leading to severe *mode collapse*.

Method Variation	BANKING77		CLINC150	
	0-shot	5-shot	0-shot	5-shot
Full Framework ($R=2$)	78.66	84.92	83.64	94.18
w/o Iterative Refresh ($R=1$)	73.50	84.22	78.98	92.78
w/o DPO (SFT Only)	71.98	83.31	76.04	91.31
w/o Hard Negatives (Random)	70.84	82.52	81.36	93.84
w/o Confusability Weighting (Uniform)	76.72	84.77	82.00	93.87
w/o MMR (Top- ℓ Substitute)	69.61	81.53	72.98	91.05

Table 3: Ablation study of key components on BANKING77 and CLINC150. **Full Framework** denotes our complete method ($R=2$). Each variant removes one component: *w/o Iterative Refresh*: single-round training ($R=1$); *w/o DPO*: supervised fine-tuning (SFT) on preferred samples only; *w/o Hard Negatives*: random inter-class negatives; *w/o Confusability Weighting*: uniform budget allocation; *w/o MMR*: naive top- ℓ selection.

5.4 Analysis and Visualization

We analyze the learned semantic space and utterance diversity to assess intent refinement.

Visualization of Decision Boundaries. We visualize feature embeddings of generated utterances using t-distributed stochastic neighbor embedding (t-SNE) (van der Maaten and Hinton, 2008). For qualitative comparison, we include *unaligned prompt outputs* sampled directly from the initial generator $\pi_{\theta}^{(0)}$ under the same instruction template $\mathcal{I}(c)$, **without** iterative refresh or DPO alignment. These samples are used solely for analysis and visualization, and are excluded from accuracy comparisons.

Generator Reliability. To assess whether preference alignment improves the generator itself, we report two generator-level metrics from Sec. 3.9: Intent Fidelity Rate (IFR; higher is better) and Neighbor Confusion Rate at k (NCR@ k ; lower is better). IFR measures how often generated utterances are recognized as the target intent by an external verifier, while NCR@ k captures how often they are assigned to the most confusable neighboring intents.

As shown in Table 4, the aligned generator improves label faithfulness (higher IFR) and reduces near-miss confusions (lower NCR@ k) compared to unaligned prompts, consistent with the clearer separation in Figure 2(b). We also use **Format Compliance Rate** (FCR) to select the refresh round R (see Appendix C.4). Additionally, alignment enhances diversity, yielding lower Self-BLEU and higher Dist-2 (Appendix C.3).

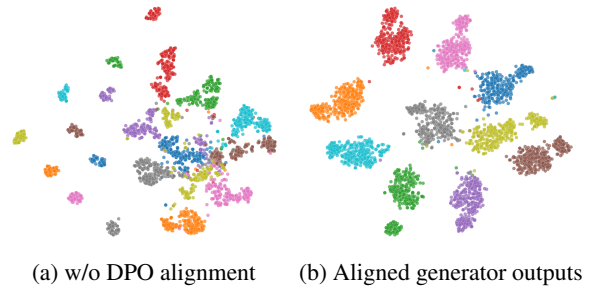


Figure 2: t-SNE visualization of generated utterances for 10 confusable intents from BANKING77 using Qwen3-8B.

Method	IFR $\uparrow$	NCR@3 $\downarrow$	NCR@5 $\downarrow$
w/o DPO alignment	0.7922	0.1469	0.2078
Aligned generator outputs	0.9804	0.0065	0.0196

Table 4: Generator reliability on BANKING77 using Qwen3-8B. Metrics are computed on **100 generated utterances per intent** (7,700 utterances in total). Higher IFR and lower NCR@ k indicate more reliable intent generation. Preference alignment substantially improves intent fidelity while sharply reducing confusion with neighboring intents.

6 Conclusion

We proposed a preference-aligned synthesis framework for fine-grained intent detection under limited supervision. The method combines multi-centroid intent prototypes, diversity-aware positive mining, confusability-aware hard negatives, and DPO with iterative preference refresh to improve synthetic data generation at the parameter level. Results on three benchmarks show consistent improvements in zero-shot and few-shot settings, while remaining competitive in fully supervised settings. We further show that the framework improves generator reliability and generalizes across multiple LLM backbones, indicating robust benefits beyond a single model family.

Limitations

Despite the empirical success of our preference-aligned synthesis framework, several limitations warrant future investigation. First, the pipeline introduces a complexity trade-off through multiple coupled hyperparameters—ranging from clustering centroids and MMR diversity thresholds to DPO regularization terms—which currently rely on empirical tuning; future work could explore adaptive parameterization to reduce manual configuration across varying taxonomies. Second, compared to

inference-only prompting baselines, our iterative “generate-mine-train” curriculum inevitably incurs higher computational overhead, presenting a cost-quality trade-off that may be constrained in low-latency deployment environments. Finally, in the generalized zero-shot setting, the framework remains sensitive to the quality of the initial candidate pool; if the base model fails to ground semantically obscure label descriptors during the cold start, these distributional biases may propagate through the anchor set, suggesting a need for more robust initialization strategies such as retrieval-augmented grounding or hybrid warm-up phases.

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 62206038), the Strategic Priority Research Program of the Chinese Academy of Sciences (No. XDA0490301), Liaoning Binhai Laboratory Project (No. LBLF-2023-01), Xiaomi Young Talents Program, and the Interdisciplinary Institute of Smart Molecular Engineering, Dalian University of Technology. We also would like to thank Shanghai Mosuan Guichao Technology Company for providing inclusive computing power and technical support.

References

- Afra Amini, Tim Vieira, and Ryan Cotterell. 2024. [Direct preference optimization with an offset](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 9954–9972.
- Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, and 32 others. 2022. [Constitutional ai: Harmlessness from ai feedback](#). *Preprint*, arXiv:2212.08073.
- Jaime Carbonell and Jade Goldstein. 1998. [The use of mmr, diversity-based reranking for reordering documents and producing summaries](#). In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’98, page 335–336, New York, NY, USA. Association for Computing Machinery.
- Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. [Efficient intent detection with dual sentence encoders](#). In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.
- Canyu Chen and Kai Shu. 2023. [PromptDA: Label-guided data augmentation for prompt-based few shot learners](#). In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 562–574, Dubrovnik, Croatia. Association for Computational Linguistics.
- Peng Chen and Bang Wang. 2025. [On collaborating small and large models for few-shot intent detection](#). In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 13937–13953, Suzhou, China. Association for Computational Linguistics.
- Yun-Nung Chen, Dilek Hakkani-Tür, and Xiaodong He. 2016. [Zero-shot learning of intent embeddings for expansion by convolutional deep structured semantic models](#). In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6045–6049.
- Paul F Christiano, Jan Leike, Tom Brown, Miljan Maric, Shane Legg, and Dario Amodei. 2017. [Deep reinforcement learning from human preferences](#). In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.
- Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. [Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces](#). *Preprint*, arXiv:1805.10190.
- Chandra Kiran Evuru, Sreyan Ghosh, Sonal Kumar, Ramaneswaran S, Utkarsh Tyagi, and Dinesh Manocha. 2024. [CoDa: Constrained generation based data augmentation for low-resource NLP](#). In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3754–3769, Mexico City, Mexico. Association for Computational Linguistics.
- Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. [Model-agnostic meta-learning for fast adaptation of deep networks](#). In *Proceedings of the 34th International Conference on Machine Learning*, pages 1126–1135.
- Biyang Guo, Yeyun Gong, Yelong Shen, Songqiao Han, Hailiang Huang, Nan Duan, and Weizhu Chen. 2022. [Genius: Sketch-based language model pre-training via extreme and selective masking for text generation and augmentation](#). *Preprint*, arXiv:2211.10330.
- Matthew Henderson, Iñigo Casanueva, Nikola Mrkšić, Pei-Hao Su, Tsung-Hsien Wen, and Ivan Vulić. 2020. [ConveRT: Efficient and accurate conversational representations from transformers](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2161–2174, Online. Association for Computational Linguistics.
- Jiwoo Hong, Noah Lee, and James Thorne. 2024. [ORPO: Monolithic preference optimization without](#)

- reference model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11170–11189, Miami, Florida, USA. Association for Computational Linguistics.
- Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. [LoRA: Low-rank adaptation of large language models](#). In *International Conference on Learning Representations*.
- Akbar Karimi, Leonardo Rossi, and Andrea Prati. 2021. [AEDA: An easier data augmentation technique for text classification](#). In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2748–2754, Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Jaekyeom Kim, Dong-Ki Kim, Lajanugen Logeswaran, Sungryull Sohn, and Honglak Lee. 2024. [Auto-intent: Automated intent discovery and self-exploration for large language model web agents](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 16531–16541, Miami, Florida, USA. Association for Computational Linguistics.
- Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. [An evaluation dataset for intent classification and out-of-scope prediction](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.
- Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. 2023. [RLAIF vs. RLHF: Scaling reinforcement learning from human feedback with AI feedback](#). *arXiv preprint arXiv:2309.00267*.
- Xinzhe Li. 2025. [A review of prominent paradigms for LLM-based agents: Tool use, planning \(including RAG\), and feedback learning](#). In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 9760–9779, Abu Dhabi, UAE. Association for Computational Linguistics.
- I-Fan Lin, Faegheh Hasibi, and Suzan Verberne. 2024. [Generate then refine: Data augmentation for zero-shot intent detection](#). In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13138–13146, Miami, Florida, USA. Association for Computational Linguistics.
- Yen-Ting Lin, Alexandros Papangelis, Seokhwan Kim, Sungjin Lee, Devamanyu Hazarika, Mahdi Namazifar, Di Jin, Yang Liu, and Dilek Hakkani-Tur. 2023. [Selective in-context data augmentation for intent detection using pointwise V-information](#). In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1463–1476, Dubrovnik, Croatia. Association for Computational Linguistics.
- Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2021. [Benchmarking Natural Language Understanding Services for Building Conversational Agents](#), pages 165–183. Springer Singapore, Singapore.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. [Roberta: A robustly optimized bert pretraining approach](#). *Preprint*, arXiv:1907.11692.
- Ilya Loshchilov and Frank Hutter. 2019. [Decoupled weight decay regularization](#). In *Proceedings of the 7th International Conference on Learning Representations (ICLR 2019)*. OpenReview.net.
- Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegrefe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. [Self-refine: Iterative refinement with self-feedback](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 46534–46594. Curran Associates, Inc.
- Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tur. 2020. [Dialoglue: A natural language understanding benchmark for task-oriented dialogue](#). *Preprint*, arXiv:2009.13570.
- Nathan Ng, Kyunghyun Cho, and Marzyeh Ghassemi. 2020. [SSMBA: Self-supervised manifold based data augmentation for improving out-of-domain robustness](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1268–1283, Online. Association for Computational Linguistics.
- Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. [Facebook FAIR’s WMT19 news translation task submission](#). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 314–319, Florence, Italy. Association for Computational Linguistics.
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. [Training language models to follow instructions with human feedback](#). In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

- Parth Patwa, Simone Filice, Zhiyu Chen, Giuseppe Castellucci, Oleg Rokhlenko, and Shervin Malmasi. 2024. [Enhancing low-resource LLMs classification with PEFT and synthetic data](#). In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 6017–6023, Torino, Italia. ELRA and ICCL.
- Cheng Qian, Bingxiang He, Zhong Zhuang, Jia Deng, Yujia Qin, Xin Cong, Zhong Zhang, Jie Zhou, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2024. [Tell me more! towards implicit user intention understanding of language model driven agents](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1088–1113, Bangkok, Thailand. Association for Computational Linguistics.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. [Direct preference optimization: Your language model is secretly a reward model](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741. Curran Associates, Inc.
- Gaurav Sahu, Olga Vechtomova, Dzmitry Bahdanau, and Issam Laradji. 2023. [PromptMix: A class boundary augmentation method for large language model distillation](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5316–5327, Singapore. Association for Computational Linguistics.
- Ziyi Shou, Yuxin Jiang, and Fangzhen Lin. 2022. [AMR-DA: Data augmentation by Abstract Meaning Representation](#). In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3082–3098, Dublin, Ireland. Association for Computational Linguistics.
- Jake Snell, Kevin Swersky, and Richard Zemel. 2017. [Prototypical networks for few-shot learning](#). In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.
- Jiakai Tang, Shiqi Shen, Zhipeng Wang, Gong Zhi, Xueyang Feng, Zexu Sun, Haoran Tan, and Xu Chen. 2025. [KAPA: A deliberative agent framework with tree-structured knowledge base for multi-domain user intent understanding](#). In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 6150–6166, Vienna, Austria. Association for Computational Linguistics.
- Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, and 179 others. 2024. [Gemma 2: Improving open language models at a practical size](#). *Preprint*, arXiv:2408.00118.
- Llama Team. 2024. [The llama 3 herd of models](#). *CoRR*, abs/2407.21783.
- Laurens van der Maaten and Geoffrey Hinton. 2008. [Visualizing data using t-sne](#). *Journal of Machine Learning Research*, 9(86):2579–2605.
- Zaitian Wang, Jinghan Zhang, Xinhao Zhang, Kunpeng Liu, Pengfei Wang, and Yuanchun Zhou. 2025. [Diversity-oriented data augmentation with large language models](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 22265–22283, Vienna, Austria. Association for Computational Linguistics.
- Jason Wei and Kai Zou. 2019. [EDA: Easy data augmentation techniques for boosting performance on text classification tasks](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.
- Henry Weld, Xiaoqi Huang, Siqu Long, Josiah Poon, and Soyeon Caren Han. 2022. [A survey of joint intent detection and slot filling models in natural language understanding](#). *ACM Comput. Surv.*, 55(8).
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. [Qwen3 technical report](#). *Preprint*, arXiv:2505.09388.
- Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-hsuan Sung, Brian Strope, and Ray Kurzweil. 2020. [Multilingual universal sentence encoder for semantic retrieval](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 87–94, Online. Association for Computational Linguistics.
- Jiacheng Ye, Jiahui Gao, Qintong Li, Hang Xu, Jiangtao Feng, Zhiyong Wu, Tao Yu, and Lingpeng Kong. 2022. [ZeroGen: Efficient zero-shot learning via dataset generation](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11653–11669, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woomyoung Park. 2021. [GPT3Mix: Leveraging large-scale language models for text augmentation](#). In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2225–2239, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Feng Zhang, Wei Chen, Fei Ding, Meng Gao, Tengjiao Wang, Jiahui Yao, and Jiabin Zheng. 2024. [From discrimination to generation: Low-resource intent detection with language model instruction tuning](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10167–10183, Bangkok, Thailand. Association for Computational Linguistics.

Feng Zhang, Wei Chen, Meng Gao, Fei Ding, Tengjiao Wang, Jiahui Yao, and Jiabin Zheng. 2025. [Clear up confusion: Iterative differential generation for fine-grained intent detection with contrastive feedback](#). In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 2207–2221, Abu Dhabi, UAE. Association for Computational Linguistics.

Jianguo Zhang, Trung Bui, Seunghyun Yoon, Xiang Chen, Zhiwei Liu, Congying Xia, Quan Hung Tran, Walter Chang, and Philip Yu. 2021. [Few-shot intent detection via contrastive pre-training and fine-tuning](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1906–1912, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jianguo Zhang, Kazuma Hashimoto, Wenhao Liu, Chien-Sheng Wu, Yao Wan, Philip Yu, Richard Socher, and Caiming Xiong. 2020. [Discriminative nearest neighbor few-shot intent detection by transferring natural language inference](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5064–5082.

A Baselines and Settings

A.1 Baseline Groups

Following the benchmark protocol in prior work (Zhang et al., 2025), we compare against 19 representative baselines spanning three groups:

(1) Specialized Low-Resource Intent Classification Models. RoBERTa (Liu et al., 2019), USE (Yang et al., 2020), CONVERT (Henderson et al., 2020), USE+CONVERT (Casanueva et al., 2020), CONVBERT (Mehri et al., 2020), CONVBERT+MLM (Mehri et al., 2020), DNNC (Zhang et al., 2020), and CPFT (Zhang et al., 2021).

(2) Traditional Text Augmentation Methods. BackTrans (Ng et al., 2019), EDA (Wei and Zou, 2019), AEDA (Karimi et al., 2021), AMR-DA (Shou et al., 2022), SSMBA (Ng et al., 2020), and GENIUS (Guo et al., 2022).

(3) Recent LLM-Based Generation/Augmentation Methods. ZeroGen (Ye et al., 2022), CoDa (Evuru et al., 2024), PromptMix (Sahu et al., 2023), ICDA-L (Lin et al., 2023), and IDG-CF (Zhang et al., 2025).

A.2 Unified Evaluation Protocol and Fairness Controls

We adopt the same evaluation protocol as (Lin et al., 2023) and report accuracy for zero-shot, few-shot (k -shot per intent), and full-shot settings. For few-shot, we sample k labeled instances per intent from the original training split (same sampling strategy as the main paper), and we keep dev/test splits unchanged.

Shared Downstream Classifier. To ensure a controlled comparison, we use RoBERTa-base (Liu et al., 2019) as the common discriminative backbone whenever the baseline permits. For baselines that explicitly require a different classifier in their original designs, we follow their standard settings. In particular, ICDA-L mandates RoBERTa-large in its original configuration, and we keep this choice to faithfully reproduce its reported setting (Lin et al., 2023).

Synthetic Data Budget Alignment. For any baseline that produces synthetic utterances for downstream training, we strictly align the per-class synthetic budget to match the benchmark protocol (Zhang et al., 2025). Concretely, we cap the number of selected synthetic instances to $N=100$ per intent (unless a baseline intrinsically defines a different regime), ensuring that performance gains are not attributable to larger synthetic corpora.

A.3 Baseline-Specific Notes

We briefly summarize baseline-specific settings that affect reproducibility:

- **Encoder-only baselines** (e.g., RoBERTa, USE, CONVERT variants, DNNC, CPFT): trained and evaluated under the same few-shot sampling protocol and metric.
- **Augmentation baselines** (e.g., BackTrans, EDA, AMR-DA, SSMBA, GENIUS): generated texts are merged with the original labeled set, and the downstream classifier is trained under the same protocol as our method, with the same per-class budget constraint when applicable.
- **LLM-based augmentation baselines** (e.g., ZeroGen, CoDa, PromptMix, ICDA-L, IDG-CF): we follow each method’s prompting/training pipeline and preserve their mandatory backbones/settings; downstream training

is still performed under the benchmark evaluation protocol, with budget normalization when applicable.

B Full Implementation Details

B.1 Fair Comparison Setup (Aligned with Prior Protocol)

Downstream Encoder Choice for Fairness. To ensure fair comparison, we fine-tune **RoBERTa-base** (Liu et al., 2019) for intent classification by adding a linear classification head, and apply the same-sized encoder to baselines whenever applicable. For **RoBERTa-Large** and **ICDA-L**, we use **RoBERTa-large** as required by their standard configurations.

Downstream Optimization. We train the downstream classifier with AdamW (Loshchilov and Hutter, 2019), using learning rate 1×10^{-5} , batch size 16, and 40 epochs.

B.2 Generator, Iterative Refresh, and Decoding Hyperparameters

Generator Backbone. We use **Qwen3-8B** (Yang et al., 2025) as the base generator.

Iterative Refresh Budget. We run iterative optimization for $R=2$ refresh rounds. In each round, we generate $B=500$ utterances per intent. With block size $\ell=4$ (below), this corresponds to $a=B/\ell=125$ blocks per intent per round.

Block Size. We set the block length to $\ell=4$ utterances per generation/optimization unit.

Decoding Hyperparameters (Non-Thinking Mode). For block-conditioned candidate generation, we use stochastic sampling in the **non-thinking** mode. Unless otherwise specified, we set temperature $T=1.0$ and nucleus sampling top- $p=0.9$.

Final Synthetic Budget. For downstream training, we select $N=100$ utterances per intent from the final-round candidate pool. The downstream budget $N = 100$ is distinct from the preference-tuple budget $M = 20$ and the per-source hard-negative retrieval depth $Q = 100$.

B.3 Preference Mining and Dataset Construction Hyperparameters

Anchor Set and Prototypes. We maintain an anchor set with a cap $K_a=5$ anchors per intent. We

construct multi-centroid prototypes with $k=5$ centroids per intent. At initialization ($r = 0$), each intent contains only its descriptor anchor and therefore uses one centroid; the configured $k = 5$ is applied from the first refreshed anchor set ($r = 1$) onward.

MMR for Diversity-Aware Selection. We use Maximal Marginal Relevance (MMR) (Carbonell and Goldstein, 1998) with trade-off $\lambda=0.7$ for diversity-aware selection, including (i) packing preferred blocks and (ii) anchor pruning / diversity control.

Preference Tuples per Intent. We construct exactly $M = 20$ block-level preference tuples per target intent and refresh round. Each tuple consists of one preferred block and one rejected block, both containing $\ell = 4$ utterances. Therefore, each target intent requires $M\ell = 80$ preferred and 80 rejected utterance slots per refresh round. The resulting preference dataset contains $M|\mathcal{C}|$ tuples in each round.

Preferred Blocks (Positives). For each target intent, we construct exactly $M = 20$ preferred blocks. Each block contains $\ell = 4$ utterances, resulting in a total positive budget of $M\ell = 80$ utterance slots per intent and refresh round. Preferred utterances are selected from the intent-conditioned candidate pool using MMR with prototype similarity as the relevance score, and selected utterances are not reused across preferred blocks.

Since each candidate pool contains $B = 500$ utterances, the pool is sufficient to construct the required 80 positive utterance slots without replacement. We additionally mix mid-similarity candidates before MMR selection to reduce over-concentration on the highest-similarity region: `pos_mix_mid=True`, `pos_mid_ratio= 0.5`, `pos_high_frac= 0.2`, `pos_mid_q_low= 0.35`, and `pos_mid_q_high= 0.75`.

Rejected Blocks (Confusable Hard Negatives). For each target intent c and source intent $c' \neq c$, we first retrieve a shortlist

$$\mathcal{T}_{c' \rightarrow c}^{(r)} = \text{Top}_Q \left(\mathcal{G}_{c'}^{(r)}; s_c^{(r)} \right),$$

where $Q = 100$ is the per-source retrieval depth. The top- Q operation constructs a candidate shortlist and does not determine the final number of negatives selected from the source intent.

To construct $M = 20$ rejected blocks of length $\ell = 4$, the required negative budget is

$$L_c^- = M\ell = 80$$

utterances per target intent and refresh round. We allocate this budget across source intents according to the confusability weights $w(c' \rightarrow c)$. Specifically, the initial quota for source intent c' is

$$q_{c' \rightarrow c} = \lfloor L_c^- w(c' \rightarrow c) \rfloor.$$

The remaining slots are assigned to source intents in descending order of their fractional remainders. Each quota is capped by the available shortlist size, and any unfilled quota is redistributed to source intents with remaining candidates.

For each source intent, we select the top $q_{c' \rightarrow c}$ candidates from its shortlist according to $s_c^{(r)}(\cdot)$, and merge the selected candidates into the hard-negative reservoir $\mathcal{N}_c^{(r)}$. We then apply MMR without replacement to pack the reservoir into exactly $M = 20$ rejected blocks. Empty candidates are removed and selected negatives are not reused within the same target intent.

Confusability Weighting. We set the confusability scale to $\gamma = 0.6$ in Eq. 10. The intent-level weight determines how the total negative budget is allocated across source intents, while utterance-level prototype similarity determines which candidates are retrieved within each source intent.

B.4 High-Confidence Subset Selection for Anchor Refresh (HighConf)

At each refresh round, we construct a high-confidence subset $\mathcal{H}_c^{(r)} = \text{HighConf}(\mathcal{G}_c^{(r)}, \mathbf{P}_c^{(r)})$ to refresh anchors and prototypes (Sec. 3.3). HighConf is implemented as a two-stage procedure: lexical de-duplication followed by embedding-space diversity selection.

Target Size and Budget-Aware Completion. We set the per-intent high-confidence target size to $h=100$. If de-duplication yields fewer than h utterances while the original pool size is at least h (i.e., shortage is caused by de-duplication rather than generation budget), we **enforce** the target size by performing diversity selection on the *original* pool to obtain exactly h utterances. If the original pool itself contains fewer than h utterances due to the generation budget, we keep all available utterances.

Stage 1: Greedy De-duplication (ROUGE-L + Jaccard Gate). We apply greedy de-duplication on the candidate pool. For a candidate x , if there exists a kept utterance x' such that (i) token n -gram Jaccard similarity is at least $g=0.4$ and (ii) ROUGE-L(F1) is at least $\tau_{\text{rouge}}=0.6$, then x is treated as a near-duplicate and removed. We compute Jaccard similarity on token bigrams ($n=2$).

Stage 2: Diversity Selection via KMeans After de-duplication, if the remaining set is larger than h , we encode utterances using sentence-transformers/all-distilroberta-v1 and apply KMeans with $n_{\text{init}}=10$ and random seed 42, then select one representative per cluster (the sample closest to each centroid) until reaching h . We use embedding batch size 256 during encoding in this step.

B.5 Preference Optimization Hyperparameters

We optimize the generator using Direct Preference Optimization (DPO) (Rafailov et al., 2023) with parameter-efficient LoRA (Hu et al., 2022). We use the default DPO temperature in ms-swift, setting $\beta=0.1$.

LoRA Configuration. We apply LoRA to all-linear modules with rank $r=8$ and LoRA alpha $\alpha=32$. We train in bfloat16 precision with maximum sequence length 2048.

Optimization Hyperparameters. We use learning rate 1×10^{-4} , per-device batch size 1 with gradient accumulation steps 16 (effective batch size 16), and warmup ratio 0.05. We set rpo_alpha= 0.1. We train for 1 epoch and use the resulting checkpoint for the subsequent refresh-round pipeline.

B.6 Compute and Hardware

All stages of our pipeline (generation, preference mining, DPO/LoRA training, filtering, and downstream training) can be executed on a single NVIDIA RTX4090 GPU under our default configuration.

B.7 Summary of Hyperparameters

For clarity, we summarize the main hyperparameters used in our implementation: $R = 2$; $\ell = 4$; $B = 500$ candidate utterances per intent and round (thus $a = 125$ generated blocks); $M = 20$ block-level preference tuples per target intent and round; $Q = 100$ retrieved hard-negative candidates per

source intent; $M\ell = 80$ preferred and 80 rejected utterance slots per target intent and round; $K_a = 5$; $k = 5$; $\lambda = 0.7$; $\beta = 0.1$; and $\gamma = 0.6$.

C Additional Results

C.1 Full-Shot Results

Methods	BANK.	HWU.	CLINC.
RoBERTa-Base	92.81	91.87	96.35
RoBERTa-Large	93.70	92.13	96.80
USE	92.81	91.25	95.06
CONVERT	93.01	91.24	97.16
USE+CONVERT	93.36	92.62	97.16
CONVBERT	92.95	90.43	97.07
CONVBERT+MLM	93.44	92.38	97.11
BackTrans	93.49	92.32	96.47
EDA	93.70	92.12	96.22
AEDA	93.99	92.17	96.43
AMR-DA	93.96	92.30	96.78
SSMBA	93.95	92.39	96.56
GENIUS	93.77	92.15	96.31
ZeroGen	93.62	92.24	96.54
CoDa	93.87	92.51	96.63
ICDA-L	94.42	92.57	97.12
IDG-CF	94.84	92.97	96.70
Ours	94.19	93.03	96.84

Table 5: Intent detection accuracy (%) in the full-shot setting on three datasets.

Why Does the Margin Narrow in Full-Shot?

This behavior is consistent with our design goal. Our method targets low-resource intent detection, so we do not use real labeled data to align the generator itself; generator alignment is performed purely from mined synthetic preferences, and the real labeled set is used only to post-train the downstream classifier (Sec. 3.10). As a result, when supervision is abundant, the classifier can rely increasingly on real examples to calibrate fine-grained decision boundaries, which naturally reduces the relative contribution of synthetic pretraining. Therefore, the full-shot regime should be viewed as a diminishing-return setting for synthesis-based methods. Even so, our method remains competitive on all three datasets and does not degrade performance, suggesting that the synthetic pretraining stage is still safe and complementary under abundant supervision.

C.2 Cross-Backbone Generalization

To examine whether the proposed preference-aligned synthesis framework is tied to a specific generator family, we further evaluate it with three backbone LLMs: Qwen3-8B, Llama3-8B, and Gemma2-9B. Following the main experimental protocol, we report downstream intent classification accuracy on BANKING77 and HWU64 under the 0-shot and 10-shot settings. As shown in Table 6, our method yields consistent improvements across all backbone families and evaluation settings, suggesting that the gains are not specific to a single generator architecture.

C.3 Diversity Analysis

To complement the generator-side reliability metrics, we report lexical diversity results under the same **Qwen3-8B** on the BANKING77 setting used for Figure 2. As shown in Table 7, lower Self-BLEU and higher Dist-2 indicate better diversity.

C.4 Format Compliance Rate as a Refresh-Selection Signal

We use **Format Compliance Rate** (FCR) not only as a formatting metric, but also as a practical signal for selecting the refresh round. Recall that a generated block is counted as compliant only if it contains exactly ℓ utterances and does not include empty lines, label leakage, or extra explanations.

Empirically, we observe that **once FCR begins to decline, downstream classification performance also drops markedly**, as shown in Table 8. This makes FCR a useful early-warning signal for generator degradation during iterative refresh. The reason is that falling FCR usually reflects a broader loss of generation discipline: when the model starts to violate the requested block format, it also tends to produce noisier, less controllable, and less intent-faithful synthetic samples. These errors then propagate to the pseudo-labeled training set and weaken the downstream classifier.

As illustrated in Table 8, while FCR remains perfect from $R=0$ to $R=2$, it drops at $R=3$, which coincides with a sharp decrease in downstream accuracy (from 83.86 to 76.16). This consistent trend further supports FCR as an effective indicator of over-refresh.

Therefore, in our pipeline, FCR is used together with generator-side reliability metrics such as IFR and NCR@k to determine whether additional re-

Backbone	BANKING77				HWU64			
	0-shot		10-shot		0-shot		10-shot	
	Vanilla	Ours	Vanilla	Ours	Vanilla	Ours	Vanilla	Ours
Qwen3-8B	74.12	78.66	86.46	90.39	72.76	80.48	80.94	88.75
Llama3-8B	73.08	78.83	87.40	90.77	72.11	80.20	80.58	88.38
Gemma2-9B	70.97	76.59	85.13	89.71	71.28	80.02	80.30	88.30

Table 6: Cross-backbone generalization results on BANKING77 and HWU64. We compare the vanilla synthesis pipeline with our preference-aligned framework using three backbone LLMs under 0-shot and 10-shot settings. Results are reported as downstream intent classification accuracy (%).

Method	Self-BLEU ↓	Dist-2 ↑
w/o DPO alignment	0.19	0.14
Aligned generator outputs	0.17	0.17

Table 7: Diversity analysis under the same **Qwen3-8B** on BANKING77 setting as Figure 2. The aligned generator achieves lower Self-BLEU and higher Dist-2, indicating reduced repetition and better lexical diversity.

Refresh Round R	FCR	IFR	Downstream Acc.
0	1.00	0.42	74.29
1	1.00	0.76	80.24
2	1.00	0.92	83.86
3	0.96	0.88	76.16

Table 8: Relationship between refresh rounds, generator reliability, and downstream performance. $R=3$ **exhibits over-refresh**: both FCR and IFR decline relative to $R=2$, accompanied by a sharp drop in downstream accuracy. This supports using generator-side reliability signals to select the stopping round for iterative refresh.

fresh rounds remain beneficial. In practice, we stop increasing R once FCR shows a clear downward trend, since further refresh tends to reduce rather than improve downstream performance.

C.5 Cold-Start Sensitivity Under Label-Only Initialization

As noted in the Limitations section, our framework is sensitive to the quality of the cold-start candidate pool in the generalized zero-shot setting. This issue becomes more evident when generation is initialized *only from label names*, without additional semantic grounding such as richer intent descriptions or external retrieval support.

In this setting, the initial generations may cover a narrower and less reliable portion of the intent manifold, especially for labels whose surface forms under-specify the true intent scope. As a result, the anchor refresh process can accumulate bias more quickly, causing the benefits of iterative pref-

erence refresh to saturate earlier than in the standard setting. Empirically, we observe that several generator-side reliability signals may deteriorate at an earlier refresh round under label-only initialization. For example, a round that remains beneficial in the standard setup can already behave like *over-refresh* in the label-only cold-start case, with metrics such as IFR and FCR beginning to decline earlier and confusion-related signals worsening correspondingly.

This earlier degradation also propagates to the downstream classifier. Because the synthetic pool becomes noisier and less boundary-faithful, the performance gain brought by iDPO is reduced, and downstream intent classification accuracy drops accordingly. These observations suggest that the optimal refresh budget is *cold-start dependent*: when initialization relies only on label names, a smaller number of refresh rounds may be preferable, and generator-side reliability signals should be monitored more conservatively to avoid premature over-iteration.

Overall, this analysis complements the main limitation discussion by showing that iDPO remains effective under label-only initialization, but its margin is smaller and its iterative training is less tolerant to aggressive refresh. This further motivates future work on stronger cold-start grounding strategies, such as enriched intent descriptions, retrieval-augmented initialization, or hybrid warm-up procedures before iterative preference optimization.

D Additional Definitions and Implementation Details

D.1 Definitions of Generator Reliability Metrics

Let $\tilde{\mathcal{D}}$ denote the generated utterance-label pairs, and let $h(\cdot)$ be the external verifier used only for evaluation.

Intent Fidelity Rate.

$$\text{IFR} = \frac{1}{|\tilde{\mathcal{D}}|} \sum_{(x,c) \in \tilde{\mathcal{D}}} \mathbf{1} \left[\arg \max_{y \in \mathcal{C}} p_h(y | x) = c \right]. \quad (13)$$

Neighbor Confusion Rate. Let $\mathcal{N}_k(c)$ denote the top- k neighbor intents of c ranked by prototype similarity. Then

$$\text{NCR@}k = \frac{1}{|\tilde{\mathcal{D}}|} \sum_{(x,c) \in \tilde{\mathcal{D}}} \mathbf{1} \left[\arg \max_{y \in \mathcal{C}} p_h(y | x) \in \mathcal{N}_k(c) \right]. \quad (14)$$

D.2 External Verifier Implementation

For generator-side evaluation, we instantiate the external verifier $h(\cdot)$ as an embedding-based nearest-label classifier. Specifically, given a generated utterance x and intent descriptors $\{d_c\}_{c \in \mathcal{C}}$, we encode both the utterance and each label descriptor using the same sentence encoder $e_{\text{ver}}(\cdot)$, and predict

$$h(x) = \arg \max_{c \in \mathcal{C}} \cos(e_{\text{ver}}(x), e_{\text{ver}}(d_c)).$$

Therefore, the verifier decision is based on sentence-embedding similarity between the generated utterance and category labels. The verifier is used only for evaluation, and is not involved in generation, anchor construction, preference mining, DPO training, or downstream classifier learning. This design provides a lightweight and label-consistent diagnostic signal without introducing an additional supervised verifier.

D.3 MMR-Based Diversity Selection

We use a single Maximal Marginal Relevance (MMR) operator (Carbonell and Goldstein, 1998) for diversity-aware selection across anchors and utterances. Given a candidate set Ω , a relevance score $\phi(\cdot)$, and a target size m , MMR constructs a subset S greedily: initialize $S = \emptyset$, then for $u = 1, \dots, m$,

$$\begin{aligned} \hat{x}_u &= \arg \max_{x \in \Omega \setminus S} \left(\lambda \phi(x) \right. \\ &\quad \left. - (1 - \lambda) \max_{x' \in S} \langle e(x), e(x') \rangle \right), \quad (15) \\ S &\leftarrow S \cup \{\hat{x}_u\}, \end{aligned}$$

where $\hat{x}_u$ denotes the element selected at the u -th greedy step. When $S = \emptyset$, the redundancy term is defined as 0. We denote the resulting set by $\text{MMR}(\Omega, \phi, m)$.

Anchor Pruning. If the anchor set $\tilde{\mathcal{A}}_c$ grows large, we prune anchors via MMR in embedding space to preserve diversity. For an anchor embedding $a \in \mathcal{A}_c$, we define its relevance as

$$\phi_{\text{anchor}}(a) = \langle a, \bar{\mathbf{p}}_c \rangle,$$

while redundancy is measured by cosine similarity $\langle a, a' \rangle$. We keep K_a anchors by

$$\mathcal{A}_c \leftarrow \text{MMR}(\mathcal{A}_c, \phi_{\text{anchor}}, K_a), \quad (16)$$

and remove or keep anchor texts in $\tilde{\mathcal{A}}_c$ according to the selected embedding indices.