Intent Detection via Vector Retrieval: a Systematic Study

Rafael Lacerda Cirolini[1] and Sandro José Rigo[1]

UNISINOS — Department of Computing
lcrafael@gmail.com

Abstract. This paper investigates a lightweight intent-detection method that removes the *fine-tuning* stage by employing vector retrieval and majority voting over the nearest neighbours. The pipeline uses nine pre-trained *sentence-transformer* models, resolves ties by the mean of the distances and applies a similarity threshold to classify *out-of-scope* (OOS) sentences. Evaluated on the *Clinc-OOS Plus* corpus, the best configuration — all-mpnet-base-v2, $k = 5$, $\tau = 0.55$ — achieved **88.1%** Top-1 accuracy, **89.1%** macro-F1, **78.9%** OOS recall and a mean latency of **18 ms**. The architecture allows new intents to be added on-line by simply appending examples to the index, without retraining.

Keywords: Intent detection · Vector retrieval · Sentence embeddings · Out-of-scope

1 Introduction

Over the last fifteen years, *intent detection* has become a key component in conversational agents, call centres and embedded voice systems. While rule trees may still suffice in some scenarios, the semantic complexity of richer interactions demands models that capture linguistic nuances, are robust to orthographic variations and can reject out-of-scope (OOS) queries. The dominant literature follows two lines: (i) *supervised classification* with *transformer fine-tuning*, and (ii) pure semantic *retrieval*, where the query is matched against previously indexed examples. The first attains high accuracy at the cost of computational expense, retraining cycles and reliance on large labelled datasets; the second is flexible but historically considered less precise.

Recent advances in *bi-encoders* such as MPNet, BGE and GTR-T5 change this picture: these models provide compact *embeddings* that preserve semantic proximity in a vector space, enabling intent detection to be treated as a k-nearest-neighbour (k-*NN*) problem followed by majority voting. This approach provides two crucial advantages for production applications:

1. **Incremental maintenance**: a new intent is added simply by inserting examples into the vector index; there is no back-propagation nor degradation on previous intents.

2. **Predictable latency**: HNSW search delivers sub-linear time $O(\log N)$, ensuring predictability even with thousands of examples.

This work systematically investigates the potential of this *no-retraining* solution. We use the *Clinc-OOS Plus* benchmark, which contains 150 in-scope intents and an OOS class, and evaluate nine *sentence-transformer* models of different sizes, including "*large*" (1024-dimensional) and *lightweight* (384-dimensional) variants. Each encoder is coupled with an identical pipeline:

1. generation of embeddings;
2. insertion into ChromaDB with an HNSW index;
3. retrieval of the k nearest neighbours;
4. decision by voting; and
5. application of a similarity threshold τ to label OOS.

The main **contributions** can be summarised in three points:

- **Comprehensive comparison** of nine models without any supervised adjustment, covering different k values $(1-20)$ and τ values $(0.30-0.75)$;
- **Specific metric** for OOS: *recall-OOS*, which quantifies the fraction of truly out-of-scope sentences correctly rejected, an aspect missing in part of the literature;
- **Operational proposal** that combines 88% accuracy with ease of maintenance, demonstrating that vector retrieval can compete with fine-tuned models in production scenarios.

The results show that the *embedding* **all-mpnet-base-v2** with $k = 5$ and $\tau = 0.55$ reaches **88.0%** Top-1 accuracy, **89.1%** macro-F1 and **78.9%** *recall-OOS*, while keeping a mean latency of **18 ms** on a T4 GPU. Moreover, we show that larger models such as BGE-Large and GTR-T5-Large approach MPNet's performance, yet with 40–70% higher latency, reinforcing the attractiveness of the base models for real-time-constrained environments.

2 Related Work

The intent-detection literature can be divided into three main strands: *supervised classification models*, *out-of-scope rejection methods* and *retrieval-based approaches.*

2.1 Supervised classification with *fine-tuning*

Since BERT [1], it has become common practice to refine *transformers* through supervised learning on datasets such as ATIS or SNIPS. Subsequent works explored larger variants (RoBERTa [2]), more compact ones (DistilBERT [3]) or models specialised in short sentences, such as TinyBERT [4]. In intent detection, Larson *et al.* [5] showed that fine-tuned BERT outperforms CNN+LSTM architectures by up to 11 p.p. in accuracy. However, these models require *full retraining* whenever new classes are introduced — a limitation mitigated by our pipeline.

2.2 Out-of-scope rejection

Detecting "out-of-domain" queries is critical for robustness. Hendrycks and Gimpel [6] proposed *maximum softmax probability* (MSP) as an uncertainty measure; Ouyang *et al.* [7] computed an energy score over BERT representations, using it to detect out-of-distribution inputs. Hou *et al.* [8] applied *meta-learning* for OOS with few examples. Our work takes a simpler route: a threshold τ on the similarity of the nearest neighbour, eliminating extra parameters.

2.3 Semantic retrieval and bi-encoders

Reimers and Gurevych [9] introduced SENTENCE-BERT, the basis for numerous bi-encoders. Karpukhin *et al.* [10] presented DPR for *open-domain QA*. More recently, MPNet [11], GTR-T5 [12] and the BGE family [13] advanced the state of the art in *retrieval*. Recent studies indicate that voting over the k nearest neighbours of sentence embeddings can reach performance comparable to fine-tuned classifiers, yet they still lack systematic evaluations of latency, OOS *recall* and hyper-parameter analysis of k and τ. Our study tackles exactly these points by (i) evaluating nine distinct bi-encoders, (ii) sweeping $k \in \{1, \ldots, 15\}$ and continuous τ, (iii) measuring end-to-end latency on production hardware and (iv) analysing OOS sensitivity.

3 Methodology

This section describes how the system obtains the intent label for an input sentence, from data preparation to runtime inference.

3.1 Dataset

We use *Clinc-OOS Plus*[1], comprising 15 150 training sentences, 3 200 validation sentences and 4 500 test sentences, covering 150 in-scope intents and one *out-of-scope* (OOS) class.

3.2 Embedding Generation and Indexing

For each pre-trained *sentence-transformer* model:

1. We encode every sentence in the training split into a dense vector (*embedding*).
2. We store *(vector, label)* pairs in an HNSW index hosted by ChromaDB. The encoder remains frozen; there is no *fine-tuning*.
3. Adding a new intent requires only inserting examples into the index, with no model retraining or HNSW graph reconstruction.

[1] Available at `https://huggingface.co/datasets/clinc_oos`.

3.3 Step-by-step Inference

Given a user sentence:

1. We generate its vector with the same encoder.
2. We query the index and retrieve the k **nearest** neighbours in terms of cosine similarity.
3. We count how many times each intent appears among those neighbours.
4. If there is a tie, we choose the intent whose neighbour set, on average, is closest to the query.
5. We inspect the distance of the closest neighbour. If this distance indicates low similarity (below the threshold τ), we label the sentence as OOS; otherwise, we return the winning intent.

3.4 Evaluated Parameters

We investigate five neighbourhood sizes ($k = \{1, 3, 5, 10, 20\}$) and four threshold values ($\tau = \{0.40, 0.55, 0.70, 0.75\}$), totalling 180 combinations per model.

3.5 Metrics

- **Top-1 accuracy**: proportion of correct predictions.
- **Macro-F1**: average F1 of each class, useful on unbalanced sets.
- **Recall-OOS**: fraction of truly OOS sentences that the system identified as such.
- **Latency**: mean time, in milliseconds, to process a query (encoding + search + decision).

3.6 Implementation and Environment

Experiments were run on Google Colab using a Tesla T4 GPU (16 GB VRAM) and 12 GB of system RAM. Main libraries: `PyTorch 2.1`, `sentence-transformers 2.7` and `chromadb 0.4`. All queries use the default HNSW index with `efSearch = 128`.

4 Results and Discussion

4.1 Overall Performance

Table 1 presents the five best configurations among the 1 620 tested (9 models × 5 k × 4 τ).

Accuracy versus simplicity. Without any supervised adjustment, MPNet surpasses 88% accuracy, showing that off-the-shelf bi-encoders, combined with vector search and voting, compete with fine-tuned approaches.

Table 1. Five best results obtained. Lat. = average latency in ms.

Model	k	τ	Acc	Macro-F1	Rec-OOS	Lat.
MPNet	5	0.55	**0.8805**	0.8911	0.789	18.1
MPNet	10	0.55	0.8789	0.8910	0.774	18.1
MPNet	3	0.55	0.8776	0.8868	**0.797**	18.1
GTR-T5-Large	5	0.70	0.8773	0.8903	0.703	31.6
BGE-Large	3	0.70	0.8755	**0.9005**	0.595	26.8

Influence of k. The gain from $k = 3$ to $k = 5$ is notable (+0.3 p.p. accuracy); $k > 10$ brings no benefit and increases ties and latency.

Impact of the threshold τ. Higher thresholds (0.70–0.75) increase OOS *recall*, but sacrifice 2–3 p.p. accuracy. The value 0.55 balances 88% accuracy and 79% OOS *recall*.

Latency. 768-dimension models (MPNet, BGE-base) remain in the 18 ms range, meeting real-time requirements. “Large” variants demand 26–32 ms, still acceptable ($<$ 300 ms in conversation) but with extra memory cost.

Incremental adherence. Adding a new intent requires only inserting vectors into the index, with no retraining or recalibration — an operational advantage over fine-tuning-based methods.

4.2 Summary

The MPNet scenario ($k = 5$, $\tau = 0.55$) offers the best balance among accuracy, OOS coverage and latency, showing that vector-retrieval techniques, coupled with a simple decision rule, are sufficient for production-grade intent-detection applications.

5 Discussion

This section delves into the main findings of the study, explores their practical implications and outlines points that require further investigation.

5.1 Simplicity *versus* Performance

The most significant result is that a *zero fine-tuning* configuration — that is, pre-trained bi-encoder, k-NN search and voting rule — reaches **88%** accuracy and **89%** macro-F1. Although the literature reports slightly higher values (90–92%) for models fine-tuned on Clinc-OOS, such gains cost hours of training, extensive hyper-parameter tuning and risk of *overfitting*. The proposed pipeline shows that, for short conversational domains, pure *retrieval* sustains competitive performance with minimal operational effort.

5.2 Operational Scalability

A distinctive attribute of the method is its **incremental scalability**. Inserting a new intent entails only:

1. labelling a few example sentences;
2. generating their vectors via the existing encoder; and
3. adding them to the index with $O(\log N)$ complexity.

There is no gradient back-propagation, index reconstruction or global retraining. In enterprise scenarios where intents evolve weekly, this *add-and-go* flow reduces dependency on machine-learning teams and enables near real-time updates.

5.3 Latency and Costs

With MPNet and $k = 5$ the average response time was **18 ms** (T4 GPU), including encoding, search and voting — below the 100 ms limit recommended for fluent dialog. "Large" variants raise latency to 26–32 ms, still within the acceptable window. Memory cost grows linearly with embedding dimension; in practice, 768 dimensions balance quality and VRAM usage (1.5 GB per model in fp16).

5.4 Limitations

- **Global threshold**. A single τ can penalise intents semantically close to OOS (e.g. *"definition"* and encyclopaedic questions). Per-intent adaptive thresholds or *class-based calibration* could mitigate this issue.
- **Persistent ties**. Even with the mean-distance tie-break, densely correlated intents (*"reminder_update", "todo_list_update"*) still show residual confusion.
- **Language dependency**. The evaluation focused on English; although multilingual models (LaBSE, Distil-MultiLM) perform satisfactorily, we did not run tests in PT-BR.

6 Conclusion and Future Work

This study showed that intent detection can be effectively treated as a vector-retrieval problem followed by voting, dispensing with *fine-tuning*. Using only pre-trained embeddings and an HNSW index, the configuration **MPNet** $+ k = 5 + \tau = 0.55$ achieved **88.0%** Top-1 accuracy and **89.1%** macro-F1 while maintaining a **mean latency of 18 ms**. Moreover, the pipeline allows new intents to be added in logarithmic time, with no retraining and no negative impact on existing intents.

Main contributions.

1. We evidenced that modern bi-encoders, used *off the shelf*, compete with fine-tuned models in accuracy but with much lower operational cost.

2. We proposed a practical *recall-OOS* metric and showed that values above 78% can be obtained with a single global threshold.
3. We provide a reproducible Colab protocol with public code and detailed latency measurements.

Future work.

Cross-encoder re-ranking. Apply lightweight models (e.g. BGE-reranker) only to the initial top-10 neighbours, with the goal of surpassing the 90% accuracy barrier without exceeding 100 ms total latency.

Class-adaptive threshold. Improve OOS detection by training a specific τ_c per intent via continuous validation, reducing false OOS on intents semantically close to the domain.

Multilingual evaluation. Replicate the experiments in Portuguese and Spanish using multilingual models (LaBSE, Distil-MultiLM) to investigate whether the same trade-off holds in morphologically distinct languages.

Cost study. Measure memory and energy consumption as the index grows to hundreds of thousands of examples, validating HNSW scalability in production.

References

1. Devlin, J., Chang, M., Lee, K., Toutanova, K. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* In: Proc. NAACL-HLT (2019). https://aclanthology.org/N19-1423
2. Liu, Y. *et al. RoBERTa: A Robustly Optimised BERT Pretraining Approach.* arXiv:1907.11692 (2019). https://arxiv.org/abs/1907.11692
3. Sanh, V., Debut, L., Chaumond, J., Wolf, T. *DistilBERT: A Distilled Version of BERT.* arXiv:1910.01108 (2019). https://arxiv.org/abs/1910.01108
4. Jiao, X. *et al. TinyBERT: Distilling BERT for Natural Language Understanding.* In: Proc. EMNLP (2020). https://arxiv.org/abs/1909.10351
5. Larson, S. *et al. An Evaluation Dataset for Intent Classification and Out-of-Scope Prediction.* In: Proc. EMNLP (2019). https://aclanthology.org/D19-1131
6. Hendrycks, D., Gimpel, K. *A Baseline for Detecting Misclassified and Out-of-Distribution Examples.* In: Proc. ICLR (2017). https://arxiv.org/abs/1610.02136
7. Ouyang, Y. *et al. Energy-based Unknown Intent Detection with Data Manipulation.* Findings of ACL (2021). https://aclanthology.org/2021.findings-acl.252
8. Hou, Y., Song, Y., Cheung, M., Zakharov, E., Fung, P. *Few-shot Intent Detection via Meta Matching Network.* In: Proc. EMNLP (2020). https://aclanthology.org/2020.emnlp-main.662
9. Reimers, N., Gurevych, I. *Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks.* In: Proc. EMNLP (2019). https://aclanthology.org/D19-1410
10. Karpukhin, V. *et al. Dense Passage Retrieval for Open-Domain Question Answering.* In: Proc. EMNLP (2020). https://aclanthology.org/2020.emnlp-main.550
11. Song, K. *et al. MPNet: Masked and Permuted Pre-training for Language Understanding.* In: Proc. NeurIPS (2020). https://arxiv.org/abs/2004.09297

12. Ni, J. *et al. Large Dual Encoders Are Generalisable Retrievers.* arXiv:2112.07899 (2021). `https://arxiv.org/abs/2112.07899`
13. Liu, Z. *et al. BGE: Beijing Academy of Artificial Intelligence General Embedding Models*. Model card and results. `https://huggingface.co/BAAI/bge-large-en`