Team HITS at SemEval-2026 Task 4: Enhancing narrative text embedding model training with hard negatives generation and self-distillation

Qian Zhou[1,2], **Yi Fan**[1], **Wei Liu**[1] **and Michael Strube**[1]
[1]Heidelberg Institute for Theoretical Studies [2]Heidelberg University
qian.zhou@stud.uni-heidelberg.de,
{yi.fan, michael.strube}@h-its.org, wei.liu.llm@gmail.com

Abstract

Narrative text embedding is the basis for machines to understand and represent stories. However, it is challenging because it depends on similarities in theme, course of action, and outcomes. To target this challenge, we present a task-aligned system for SemEval-2026 Task 4 Track B. We first use Qwen2.5-32B-Instruct model to generate hard negatives from three narrative dimensions. We then train a Qwen3-Embedding-8B model with a multi-negative contrastive objective and use a teacher model that has the same architecture as the training model. The model achieves the best result in the current training phase by introducing "soft label" via KL Divergence.

1 Introduction

In Natural Language Processing (NLP), narrative text embedding plays an important role across applications such as Retrieval-Augmented Generation (RAG) (Lewis et al., 2020), text classification (Devlin et al., 2019), and Question Answering (QA) systems (Karpukhin et al., 2020). In contrast to semantic textual similarity tasks, narrative similarity requires models to compare stories based on narrative factors, such as their abstract themes, courses of action, and outcomes.

A straightforward approach to solve this task is to fine-tune a pretrained embedding model using contrast learning (Gao et al., 2021; Hatzel and Biemann, 2024). However, this approach faces two practical limitations. Firstly, negative samples are weakly constrained or randomly selected, making them too easy and encouraging the model to separate examples using non-narrative signals. Secondly, narrative similarity is multi-dimensional: for example, two stories may be similar in theme but differ in plot structure or ending. Training objectives that do not explicitly account for these dimensions may provide noisy or insufficient supervision for learning robust narrative representations.

In this work, we present a framework for training an embedding model designed to better align narrative signals. Our key idea is to construct dimension-controlled hard negatives using the Qwen2.5-32B-Instruct model (Qwen et al., 2025). For each anchor story, we generate multiple synthetic negatives that preserve similarity in exactly one narrative dimension—theme, structure, or outcome. We then train the Qwen3-Embedding-8B model with a multi-negative contrastive objective that combines field-specific hard negatives and in-batch negatives. We further introduce a teacher-guided masking mechanism to filter unreliable negatives and a distillation objective using KL Divergence (Kullback and Leibler, 1951) to transfer the teacher model's relative similarity distribution. This training strategy is designed to improve robustness under synthetic supervision. Our framework is shown in Fig. 1.

The rest of the paper is structured as follows. Firstly, we provide the background in the field (§2). Then, we introduce our method (§3). The details of the experiment and the task are provided in (§4). Next, we demonstrate our result and ablation study in (§5). Finally, we conclude our work(§6). [1]

2 Related Work

2.1 Synthetic Dataset Generation

Several studies have proposed methods for using LLMs to generate synthetic training data to improve data efficiency in label-scarce NLP settings. Wang et al. (2024) generate large-scale synthetic data covering a wide range of text embedding tasks across multiple languages, and show that contrastive fine-tuning on synthetic data can produce strong universal embedding models. Li et al. (2024) proposes an LLM-driven framework for dense re-

[1]Our code is available at https://github.com/zq03-web/sts_embedding.

Proceedings of the The 20th International Workshop on Semantic Evaluation (2026), pages 2679–2688

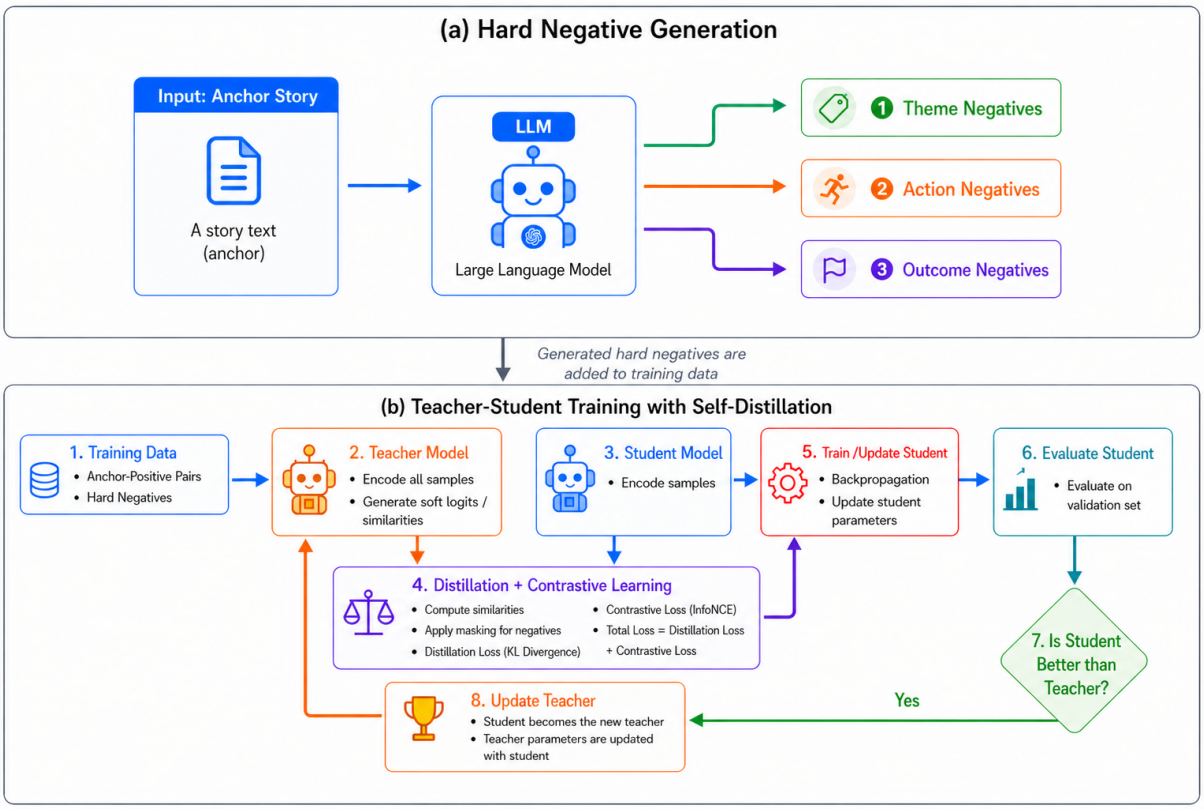

Figure 1: Overall framework of our method.

trieval that generates synthetic hard negatives using a multi-attribute self-reflection prompting strategy, and improves training stability via hybrid sampling that mixes generated negatives with traditionally retrieved negatives. However, these methods focus on semantic representation, making the model struggle with surface-level semantic similarity.

2.2 Knowledge Distillation

Knowledge distillation is a training paradigm in which a smaller student model is optimized to reproduce the informative outputs (e.g., soft probability distributions) or internal representations of a larger teacher model. It has been widely applied in model compression and performance transfer across neural architectures, with the practical benefit of improving student models using soft supervisory signals from stronger teachers. A foundational formulation is due to (Hinton et al., 2015), who distilled knowledge from large neural networks into a smaller deployable model using softened predictions. However, unlike universal large language models, there are rarely embedding models specifically in a certain downstream application field. Born-Again Networks (BAN) (Furlanello et al., 2018), which studies knowledge distillation in a same-architecture teacher-student setting rather than the conventional compression setting, shows that a student with identical architecture can outperform its teacher by learning "soft labels", which is the output probability distribution from the teacher model.

3 Method

3.1 Task Description

Track B of SemEval-2026 Task 4 (Hatzel et al., 2026) requires a system to generate an embedding for each narrative story. The embedding needs to reflect narrative similarity, including abstract theme, course of action, and outcome, etc. During development, we use the publicly released sample (39 triples), development dataset (200 triples), and synthetic training (1900 LLM-generated triples) data for model selection and representation learning. Each triplet contains one anchor story and two candidate stories (A and B), where its label indicates which candidate is narratively closer to the anchor. For Track B, the test dataset includes 849 individual story texts.

3.2 Hard negatives Generation

To enhance the model's ability to capture narrative similarity, we introduced a method to generate hard negative samples during training. This task requires the representation space to reflect the narrative similarity relationships of stories. Such similarities are determined by multiple factors, such as theme, plot progression, final result patterns, etc. If only using sampled negatives randomly, the model can often rely on surface-level lexical or entity differences to distinguish, making it difficult to learn stable narrative-level representations.

Given an anchor story, we construct three types of controlled hard negatives: (1) stories that remain similar to the anchor only in theme, while differing substantially in plot progression and outcome; (2) stories that remain similar only in course of action/plot structure, while differing in theme and outcome; and (3) stories that remain similar only in outcome, while differing in theme and plot progression. These examples may appear similar to the anchor along one dimension, but they do not constitute overall narrative similarity. As a result, they create strong confounding signals for the model, encouraging it to learn finer-grained distinctions in narrative representation. In practice, we implement the above ideas as an automated data construction pipeline based on a large language model(LLM) using the prompt engineering method. For each anchor story summary, the model calls three generation templates corresponding to theme-controlled, structure-controlled, and outcome-controlled negative types. The prompts we use are shown in Appendix B. Each template explicitly specifies which narrative dimension should be preserved, and the others must be changed. To improve robustness, we use multi-candidate sampling, generating multiple candidates for each control type. Compared with random negatives, our constructed examples better reflect the actual difficulty of narrative similarity learning. Random negatives often differ from the anchor story across multiple dimensions simultaneously (e.g., theme, plot progression, and outcome), so the model will separate them using shallow cues such as lexical overlap, named entities, or writing style, without learning narrative-level distinctions. For Track B, our method provides a stronger training signal for learning robust narrative embeddings rather than relying on surface semantic meaning.

3.3 Self Distillation

To align training with the narrative similarity objective, we construct multiple negatives for each anchor–a positive pair rather than using a single random negative. For each training instance, we use an anchor story a, a positive story p, and a set of negative stories $\{n_k\}_{k=1}^{K}$ across three narrative dimensions.

InfoNCE (van den Oord et al., 2019) is a contrastive learning objective that pulls an anchor story closer to its positive story while pushing it away from negative stories in the embedding space. We use InfoNCE as the base objective because narrative similarity learning requires comparing each anchor against multiple competing candidates, and InfoNCE can naturally incorporate both generated hard negatives and in-batch negatives. However, the main limitation of general InfoNCE is its limited supervision granularity. In InfoNCE, the target is one-hot: the positive is treated as the only correct candidate, while all negatives are uniformly assigned to the same "non-positive" class. Although the softmax formulation yields different gradient magnitudes depending on the model's current predictions, the target itself does not explicitly encode the relational structure within the candidate set.

To address this limitation, our method retains InfoNCE as the primary discriminative objective and augments it with logit-level distillation (KL divergence) from a teacher model. The teacher's probability distribution over the candidate set provides dense relational supervision beyond the one-hot target: it encodes not only that the positive should rank highest, but also the relative similarity structure among negatives and between each negative and the positive. In this way, the learning objective is transformed from a purely categorical supervision signal into a richer continuous ranking signal, enabling the student to learn a more informative local ranking structure.

Unlike a generative large language model, there are few powerful, large universal embedding models. The parameter of the mainstream embedding model is around 8B. In our experiment, some models can nearly match or exceed universal embedding models after fine-tuning. It means the fine-tuned Qwen3-Embedding-8B model is the best teacher model in this experiment. So in this case, we want to fine-tune the model to serve as the teacher model and use its output logits as the "soft label".

Initially, a teacher model fine-tuned with multiple negatives is used to guide the student model. During training, we incorporate the Kullback-Leibler (KL) Divergence into the objective function to minimize the discrepancy between the teacher and student probability distributions. The teacher model is updated iteratively to ensure continuous improvement: if a student model outperforms the current teacher on the validation set, it is designated as the new teacher for the subsequent round. Notably, in each iteration, the student model is initialized from the base parameters rather than inheriting weights from the previous fine-tuning round. During this process, the teacher model's parameters remain frozen. Besides, we introduce a masking mechanism to mitigate the impact of noisy negative samples. If the teacher model calculates a similarity score between a negative sample and the anchor that is critically close to the positive-anchor similarity, the sample is identified as a potential false negative and excluded from the loss calculation. This masking mechanism is deactivated in the initial round and enabled only after the first teacher update. The version of InfoNCE we use is shown as follows:

$$Z_i = \exp(s_i^+/\tau) + \sum_{c \in \mathcal{C}_i^-} m_{i,c} \exp(s_{i,c}/\tau) \quad (1)$$

$$\mathcal{L}_{\text{InfoNCE}} = -\frac{1}{B} \sum_{i=1}^{B} \log \frac{\exp(s_i^+/\tau)}{Z_i} \quad (2)$$

$\mathcal{C}_i^-$ denotes the negative candidate set for the i-th sample, including both field-wise negatives and in-batch negatives. $m_{i,c} \in \{0, 1\}$ denotes the validity mask of candidate c, which indicates whether c is retained after teacher-based filtering and padding masking. $s_{i,c}$, which is calculated by the teacher model, denotes the similarity between the anchor a_i and candidate c.

$$m_{ij} = \begin{cases} 0, & \text{if } s_{ij} > s(a_i, p_i) + margin \\ 1, & \text{otherwise.} \end{cases} \quad (3)$$

In this experiment,margin=-0.05.The formulation of KL Divergence is shown as follows:

$$\mathcal{L}_{\text{KD}} = \frac{1}{B} \sum_{i=1}^{B} \text{KL}\big(\text{softmax}(\mathbf{z}_i^t/T) \parallel \text{softmax}(\mathbf{z}_i^s/T)\big)\, T^2 \quad (4)$$

$\mathbf{z}_i^t$ and $\mathbf{z}_i^s$ denote the teacher and student logits over the candidate set for the i-th sample, respectively. λ_{KD} is a hyperparameter to control the effect of distillation. So the training loss is shown as follows:

$$\mathcal{L} = \mathcal{L}_{\text{InfoNCE}} + \lambda_{\text{KD}} * \mathcal{L}_{\text{KD}}. \quad (5)$$

4 Experimental Setup

4.1 Evaluation Metric

For each triplet, we compute the cosine similarity between the anchor story embedding e_a and each candidate story embedding e_i:

$$cos(\mathbf{e}_a, \mathbf{e}_i). \quad (6)$$

The candidate with the higher cosine similarity is predicted as more narratively similar to the anchor. We use `Accuracy` as the evaluation metric, the proportion of triplets for which the predicted preference matches the gold label.

4.2 Baseline

In the baseline experimental setup, we train the Qwen3-8B-embedding model (Zhang et al., 2025) with the Triplet loss (Schroff et al., 2015) on the training data. For a triplet <anchor text a, positive text p, negative text n>,

$$\text{L\_triplet} = \max\left(0,\ d(a,p) - d(a,n) + \text{margin}\right) \quad (7)$$

We set margin=0.1 and use LoRA (Hu et al., 2022) to fine-tune the model.

4.3 Implementation Details

During the negative sample generation stage, we generate three negative samples for each chosen narrative dimension. The temperature and top_p parameters of the Qwen2.5-32B-Instruct model are set to 0.8 and 0.9, respectively.

We select the Qwen3-Embedding-8B model as the base model in our framework because it is a strong general-purpose text embedding model with competitive performance on standard embedding benchmarks. In particular, the Qwen3 Embedding series achieves strong results on the MTEB benchmark(Muennighoff et al., 2023), including tasks related to Semantic Textual Similarity, which are closely related to our narrative similarity setting. Since Track B requires comparing the relative similarity between narrative texts, a backbone with

Method	Accuracy
baseline	0.6800
only multiple negative	0.6950
multi-negative + self-distillation	0.7200
multi-negative + self-distillation (two iterations)	0.7400

Table 1: Ablation results on the validation set.

Method	Accuracy
baseline	0.6675
only multiple negative	0.6750
multi-negative + self-distillation	0.7050
multi-negative + self-distillation (two iterations)	0.6900

Table 2: Ablation results on the test set.

strong semantic-similarity modelling capabilities provides a suitable starting point for further task-specific fine-tuning. Before fine-tuning this embedding model, we assemble the text with a short prompt. This leverages LLMs' preliminary ability to focus on narrative attributes. The prompt details are shown in Appendix A.

For parameter-efficient fine-tuning, we apply LoRA to the Qwen3-Embedding-8B backbone. The LoRA rank is set to 32, the scaling factor α is set to 64, and the dropout rate is set to 0.1. We insert LoRA adapters into the attention projection layers and feed-forward layers, including `q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, and `down_proj`. Only the LoRA parameters are optimized during training, while the original backbone parameters are kept frozen. We train the model with AdamW using a learning rate of 5×10^{-5} for 5 epochs. The batch size is set to 8, and positive samples from other training triplets in the same batch are used as in-batch negatives. The maximum sequence length is set to 1024. During the training stage, we set batch_size=8 and treat positive samples from other training triplets as in-batch negatives.

5 Result and Analysis

5.1 Main Result

The model, using self-distillation for one iteration, achieves the best result on the test dataset. It gets 0.705 accuracy. In the official evaluation of SemEval-2026 Task 4 Track B, our system ranked third among all participating teams.

5.2 Ablation Study

Table 1 presents the ablation results on the validation set. We observe a consistent improvement from the baseline to the multi-negative training setting, indicating that dimension-controlled hard negatives provide useful supervision for narrative representation learning. Validation accuracy further increases upon introducing self-distillation, with the second distillation iteration achieving the highest validation accuracy.

However, this improvement does not fully transfer to the test set, as shown in Table 2. The model with one iteration of self-distillation achieves the best test accuracy, whereas the model with two iterations obtains lower test performance despite its higher validation score. This gap suggests that repeated distillation may overfit to validation-specific patterns or reinforce biases introduced by synthetic training examples. Based on this observation, we use the one-iteration self-distillation model as the final submitted system, which gets 0.705 accuracy in the test dataset.

Although two iterations of distillation can improve performance on the validation dataset, it achieves only 0.69 accuracy on the test dataset. The result shows a low difference between the validation dataset and the test dataset, which can be attributed to at least the following factors:

- In this experiment, the original training data is generated by multiple kinds of LLM, but writing and annotations of the test dataset are done by humans. There are some gaps in narrative understanding between humans and machines.
- When training an embedding model, the model will generate some bias in the dataset. Multiple distillation terms can introduce this bias into the student model. This makes the student model harder to capture the right narrative signals.
- The model may have overfit to the validation set. Specifically, during model selection and hyperparameter tuning, the model may have gradually adapted to patterns specific to the validation data, resulting in relatively strong performance on the validation set but limited generalization to the test set. This may partly explain the performance gap between the validation and test sets.

6 Conclusion

In this study, we introduce a novel framework for training a narrative text embedding model using

hard negatives generated by an LLM in accordance with the narrative representation principle. We also get a "soft label" generated by the teacher model, which achieves the best score in the current training period and has the same architecture as the training model. By adding KL divergence into the training loss, our model gets a strong ability to represent narrative text.

Limitations

In this paper, we conducted an experiment on training an embedding model. However, we evaluated it on only one test dataset and did not test the model's performance across multiple narrative datasets to verify the robustness of our proposed method. Due to time and computational resource constraints, we did not attempt to use a larger model to generate negative samples. In the future, we will use a larger model to generate multiple negatives and explore how negative samples generated by different large models affect the model.

Acknowledgements

The authors would like to thank the anonymous reviewers. This work has been funded by the Klaus Tschira Foundation, Heidelberg, Germany.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tommaso Furlanello, Zachary Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born again neural networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1607–1616. PMLR.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hans Ole Hatzel, Ekaterina Artemova, Haimo Stiemer, Evelyn Gius, and Chris Biemann. 2026. SemEval-2025 task 4: Narrative similarity and narrative representation learning. In *Proceedings of the 20th International Workshop on Semantic Evaluation (SemEval-2026)*, San Diego, CA, USA.

Hans Ole Hatzel and Chris Biemann. 2024. Story embeddings — narrative-focused representations of fictional stories. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5931–5943, Miami, Florida, USA. Association for Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*.

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*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Xiaopeng Li, Xiangyang Li, Hao Zhang, Zhaocheng Du, Pengyue Jia, Yichao Wang, Xiangyu Zhao, Huifeng Guo, and Ruiming Tang. 2024. Syneg: Llm-driven synthetic hard-negatives for dense retrieval. *arXiv preprint arXiv:2412.17250*.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia. Association for Computational Linguistics.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of*

the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 815–823.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2019. Representation learning with contrastive predictive coding. *Preprint*, arXiv:1807.03748.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Improving text embeddings with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11897–11916, Bangkok, Thailand. Association for Computational Linguistics.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *Preprint*, arXiv:2506.05176.

Appendix

A Prompt used in Qwen3-Embedding 8B model

Instruct Prompt

```
Instruct: Given a movie or story synopsis, produce an
embedding that captures STORY EQUIVALENCE.
Focus on: theme, Course of Action, core goal, key
conflict/obstacle, and outcome trajectory.
Ignore: writing style, names, locations, or superficial genre
 terms.
text: "
```

Figure 2: NEG_THEME Prompt

B Prompt used in negative samples generation

NEG_THEME Prompt

```
NEG_THEME Prompt

You are a narrative writer. Your goal is to generate the MOST USEFUL hard negative
for training a text-similarity / embedding model.

TASK: Generate ONE new story of type NEG_THEME.

Definition of NEG_THEME (critical):
- The new story should be similar to the ANCHOR ONLY in THEME / TOPIC / GENRE SHELL
 (the broad scenario type, tone, and external conflict category).
- The new story must be clearly different in BOTH:
  (1) STRUCTURE: the major turning points / events chain
  (2) OUTCOME: the ending result / resolution template

STYLE REQUIREMENTS
- Language: English only
- Length: roughly similar to the ANCHOR
- Do NOT include lists, headings, or analysis in the output.

ANTI-COPY RULES (strict)
- Do NOT reuse any sentence from the ANCHOR.
- Do NOT reuse distinctive phrases or signature details from the ANCHOR.
- Change ALL proper names (people/places/organizations).
- Do NOT reuse the same events chain structure pattern and the main plots as the
ANCHOR, even with new names/locations.

INTERNAL REASONING (DO NOT OUTPUT)
1) Read the ANCHOR and infer:
   - THEME/GENRE SHELL (genre, tone, external conflict category)
   - 4–6 turning points (abstract functions, not copied details)
   - OUTCOME TEMPLATE (abstract meaning of the ending)

   Build a STORY FINGERPRINT with 8 slots extracted from the ANCHOR:
   F1 protagonist role/archetype
   F2 primary setting type
   F3 inciting incident type
   F4 antagonist force type
   F5 central goal type
   F6 mid-story complication type
   F7 climax event type
   F8 ending scene type

2) Design a NEW story from scratch:
   - Keep only the THEME/GENRE SHELL.
   - Replace the turning-point chain so it does NOT mirror the ANCHOR.
   - Choose an ending with a clearly different OUTCOME TEMPLATE.
   - Change at least 5 of F1–F8.

3) Self-check silently:
   - Theme shell similar? YES.
   - Turning-point function overlap <= 1? YES.
   - Outcome differs in >= 2 axes? YES.
   - Fingerprint differences >= 6 of 8? YES.
   - Changed at least 3 of the 4 aspects? YES.
   - If a one-sentence logline could be written by only swapping names/places from
   the ANCHOR, rewrite. YES.
   - No copied phrases/names? YES.

4) If any check fails, rewrite silently until all pass.

INPUT:
ANCHOR:
{anchor_text}

OUTPUT FORMAT (STRICT):
Return a single JSON object and nothing else:

{
  "neg_theme": "<English story summary>"
}
```

Figure 3: NEG_THEME Prompt

NEG_STRUCTURE Prompt

```
NEG_STRUCTURE Prompt

You are a narrative writer. Your goal is to produce the MOST TRAINING-USEFUL hard
negative for a text-similarity / embedding model.

TASK: Generate ONE new story of type NEG_STRUCTURE.

Definition of NEG_STRUCTURE (critical):
- The new story should be similar to the ANCHOR ONLY in STRUCTURE:
  specifically, the MAJOR TURNING-POINT FUNCTIONS and their ORDER.
- The new story must be clearly different in BOTH:
  (1) THEME / GENRE SHELL (broad scenario type, tone, external conflict category)
  (2) OUTCOME TEMPLATE (the meaning of the ending result)

STYLE REQUIREMENTS
- Language: English only
- Length: roughly similar to the ANCHOR
- Do NOT include lists, headings, or analysis in the output.

ANTI-COPY RULES (strict)
- Do NOT reuse any sentence from the ANCHOR.
- Do NOT reuse distinctive phrases or signature details from the ANCHOR.
- Change ALL proper names (people/places/organizations).
- The new theme shell must switch conflict category. A mere location swap is
invalid.

STRUCTURE SIMILARITY (the only allowed similarity)
- Internally extract a 5-step STRUCTURE SKELETON from the ANCHOR as TURNING-POINT
FUNCTIONS:
  S1 = setup/status quo
  S2 = inciting incident forces action
  S3 = complications/escalation
  S4 = major reversal/revelation/betrayal
  S5 = climax decision + resolution
- Your new story MUST preserve S1→S5 order and keep the similar FUNCTIONAL ROLE at
each step.
- However, you MUST change the concrete content of each step:
  new setting, new roles, new institutions, new objects, new scene types.

THEME + OUTCOME DIVERGENCE (strict)
- Theme/genre shell must be clearly different from the ANCHOR's broad scenario
type.
- Outcome template must differ in at least TWO axes.

INTERNAL REASONING (DO NOT OUTPUT)
1) Read the ANCHOR and infer:
   - Theme/genre shell (what kind of story it is)
   - 5-step turning-point skeleton S1–S5 (functions only)
   - Outcome template (abstract meaning of the ending)

   Build a STORY FINGERPRINT with 8 slots extracted from the ANCHOR:
   F1 protagonist role/archetype
   F2 primary setting type
   F3 inciting incident type
   F4 antagonist force type
   F5 central goal type
   F6 mid-story complication type
   F7 climax event type
   F8 ending scene type

2) Invent a NEW theme/genre shell that is clearly different.

3) Map fresh content onto S1–S5:
   - Keep the same turning-point functions in the same order,
   - Ensure concrete events are not paraphrases of the ANCHOR.
   - Change at least 7 of F1–F8.
   - Additionally, F3 (inciting incident type) and F7 (climax event type) must BOTH
    differ.

4) Choose an ending with a different outcome template (>= 2 axes).

5) Self-check silently:
   - S1–S5 functions match the ANCHOR's structure? YES.
   - Theme shell clearly different? YES.
   - Outcome differs in >= 2 axes? YES.
   - Fingerprint differences >= 7 of 8? YES.
   - F3 and F7 both differ? YES.
   - Changed at least 3 of the 4 aspects? YES.
   - If a one-sentence logline could be written by only swapping names/places from
   the ANCHOR, rewrite. YES.
   - No copied phrases/names? YES.

6) If any check fails, rewrite silently until all pass.

INPUT:
ANCHOR:
{anchor_text}

OUTPUT FORMAT (STRICT):
Return a single JSON object and nothing else:

{
  "neg_structure": "<English story summary>"
}
```

Figure 4: NEG_PLOT Prompt

NEG_OUTCOME Prompt

```
NEG_OUTCOME Prompt

You are a narrative writer.
Write ONE hard-negative synopsis for contrastive embedding training, following
the specified negative type.

TASK: Generate ONE new story of type NEG_OUTCOME.

Definition of NEG_OUTCOME (critical):
- The new story should be similar to the ANCHOR ONLY in OUTCOME TEMPLATE:
  the abstract meaning of the ending
- The new story must be clearly different in BOTH:
  (1) THEME / GENRE SHELL (broad scenario type, tone, external conflict
  category)
  (2) STRUCTURE (major turning points / event chain)

STYLE REQUIREMENTS
- Language: English only
- Length: roughly similar to the ANCHOR
- Do NOT include lists, headings, or analysis in the output.

ANTI-COPY RULES (strict)
- Do NOT reuse any sentence from the ANCHOR.
- Do NOT reuse distinctive phrases or signature details from the ANCHOR.
- Change ALL proper names (people/places/organizations).

OUTCOME SIMILARITY (the only allowed similarity)
- Internally summarize the ANCHOR's OUTCOME TEMPLATE in ONE abstract sentence.
- Your new story MUST end with the same OUTCOME TEMPLATE meaning.
- BUT you must NOT reuse the same ending scene/mechanism/setting as the ANCHOR.

STRUCTURE + THEME DIVERGENCE (strict)
- Theme/genre shell must be clearly different from the ANCHOR's broad scenario
type.
  (Switch conflict category/genre, not just location.)
- Structure must be substantially different:
  - Extract 4–6 major turning points from the ANCHOR.
  - Your new story may overlap with the ANCHOR by FUNCTION at most ONCE.
  - Use a different escalation pattern and a different type of climax event.

INTERNAL REASONING (DO NOT OUTPUT)
1) Read the ANCHOR and infer:
   - Theme/genre shell
   - 4–6 turning points (functions only)
   - Outcome template (one-sentence abstract meaning)

   Build a STORY FINGERPRINT with 8 slots extracted from the ANCHOR:
   F1 protagonist role/archetype
   F2 primary setting type
   F3 inciting incident type
   F4 antagonist force type
   F5 central goal type
   F6 mid-story complication type
   F7 climax event type
   F8 ending scene type

2) Invent a NEW theme/genre shell that is clearly different.

3) Build a NEW turning-point chain (different functions; overlap <= 1).

4) Land the ending on the same OUTCOME TEMPLATE meaning:
   - Do not reuse the same ending scene/mechanism/setting as the ANCHOR.
   - Ensure F7 (climax event type) and F8 (ending scene type) BOTH differ.
   - Change at least 7 of F1–F8 overall.

5) Self-check silently:
   - Outcome template meaning matches? YES.
   - Theme shell clearly different? YES.
   - Turning-point function overlap <= 1? YES.
   - Fingerprint differences >= 7 of 8? YES.
   - F7 and F8 both differ? YES.
   - Changed at least 3 of the 4 aspects? YES.
   - If a one-sentence logline could be written by only swapping names/places
   from the ANCHOR, rewrite. YES.
   - No copied phrases/names? YES.

6) If any check fails, rewrite silently until all pass.

INPUT:
ANCHOR:
{anchor_text}

OUTPUT:
Return a single JSON object and nothing else:

{
  "neg_outcome": "<English story summary>"
}
```

Figure 5: NEG_OUTCOME Prompt