VerbaNexAI at SemEval-2026 Task 4: Two-Stage Narrative Similarity via Fine-Tuned Bi-Encoder with MLP Ensemble

Pablo A. Pertuz-Duran, Edwin Puertas, Jairo Serrano, Juan Carlos Martinez-Santos
Universidad Tecnologica de Bolivar
Cartagena, Colombia
{ppertuz,epuerta,jserrano,jcmartinezs}@utb.edu.co

Abstract

This paper describes VerbaNex AI's participation in SemEval-2026 Task 4: Narrative Story Similarity and Narrative Representation Learning (Hatzel et al., 2026), a shared task on assessing semantic relatedness between short narrative texts. The task comprises two tracks: Track A requires selecting which of two candidate stories is more similar to an anchor, and Track B requires producing fixed-size story embeddings whose cosine similarity reflects narrative relatedness. We propose a unified two-stage system built on Qwen3-Embedding-0.6B. The first stage fine-tunes the encoder as a bi-encoder with a 512-dimensional projection head using a composite loss combining margin ranking, pairwise softmax, and multiple negatives ranking objectives. The second stage trains a lightweight MLP head over frozen bi-encoder embeddings using pairwise interaction features, with k-fold cross-validation and logit-averaging ensemble inference. The system was trained exclusively on the official supervised data without leveraging the additional 1,900 synthetic triples generated by LLM released by the organizers. Although the system ranked first on both tracks in the development phase, its performance did not transfer to the official test set, where it ranked 45 on Track A and 21 on Track B.

1 Introduction

Assessing whether two narratives convey similar underlying stories is a challenging problem that extends beyond surface-level lexical overlap. Narrative similarity requires modeling events, temporal progression, and causal structure (Chambers and Jurafsky, 2008a). Unlike sentence-level semantic textual similarity (STS) (Cer et al., 2017a), narrative relatedness involves reasoning over multi-sentence texts where semantic alignment may be implicit, structurally varied, or abstract.

SemEval-2026 Task 4, Narrative Story Similarity and Narrative Representation Learning (Hatzel et al., 2026), addresses this challenge by evaluating systems on their ability to assess narrative relatedness between short story summaries. The shared task consists of two complementary tracks. Track A formulates the problem as comparative ranking: given an anchor story and two candidate stories, systems must determine which candidate is more similar to the anchor. Track B requires generating fixed-size story embeddings such that cosine similarity reflects narrative relatedness. Together, these tracks evaluate both decision-level ranking performance and the quality of learned semantic representations.

Recent advances in contrastive learning have significantly improved embedding-based similarity modeling (Gao et al., 2021a). Motivated by this progress, we develop a unified two-stage system built on `Qwen3-Embedding-0.6B`. In the first stage, we train a bi-encoder to learn narrative-level representations using contrastive objectives tailored to the task. In the second stage, we refine similarity predictions through a lightweight interaction model operating over frozen embeddings. This design allows us to balance representation quality with pairwise decision modeling while maintaining computational efficiency.

The code is publicly available at `https://github.com/VerbaNexAI`.

2 Background

2.1 Task and Dataset

SemEval-2026 Task 4 provides a dataset of short narrative texts annotated for pairwise narrative similarity (Hatzel et al., 2026). The Track A development set contains triples of the form (anchor, candidate A, candidate B), each labeled with a binary indicator of which candidate is narratively closer to the anchor. The Track B development set contains individual story texts to be embedded into a shared vector space. In addition to the labeled

Proceedings of the The 20th International Workshop on Semantic Evaluation (2026), pages 3130–3136

development data, the organizers released a synthetic training set of 1,900 triples generated using large language models, intended to supplement the small development set. Our system was trained exclusively on the development data and did not incorporate the synthetic triples.

Statistic	Value
Dev triples (Track A)	200
Dev stories (Track B)	479
Synthetic triples (LLM-gen.)	1,900
Avg. story length (tokens)	122.0

Table 1: Dataset statistics for the Narrative Similarity task.

Table 1 summarizes the dataset statistics. The development set is small (200 labeled triples), which motivated our design choices toward parameter-efficient architectures and k-fold cross-validation. The availability of 1,900 synthetic triples, which we did not use, represents a roughly 9.5$\times$ increase in training data that could have significantly improved generalization.

2.2 Related Work

Sentence-level semantic similarity has been widely studied through benchmarks such as STS Benchmark (Cer et al., 2017b) and SICK (Marelli et al., 2014). The dominant paradigm uses bi-encoder architectures trained with contrastive or ranking losses, as popularized by Sentence-BERT (Reimers and Gurevych, 2019) and subsequent models such as SimCSE (Gao et al., 2021b) and E5 (Wang et al., 2022). More recent embedding models, including BGE (Xiao et al., 2023) and Qwen3-Embedding (Team, 2025), have advanced the state of the art on MTEB benchmarks (Muennighoff et al., 2023) by leveraging larger transformer backbones and improved training objectives.

However, narrative similarity extends beyond sentence-level semantics. Prior work in computational narrative understanding has explored event sequence alignment (Chambers and Jurafsky, 2008b) and narrative structure extraction (Reagan et al., 2016). These approaches typically require explicit narrative parsing, making them difficult to integrate into an end-to-end embedding framework.

Our approach builds directly on the bi-encoder paradigm established by Sentence-BERT (Reimers and Gurevych, 2019), extending it with a task-specific composite loss and a projection head for dimensionality reduction. Unlike prior narrative analysis methods that require explicit structural representations, our system relies entirely on learned embeddings from a general-purpose encoder fine-tuned with narrative similarity annotations. Our initial experiments used BGE-Large (Xiao et al., 2023) as the backbone, but upgrading to Qwen3-Embedding-0.6B (Team, 2025) yielded substantial gains on the development set. As our test results show, however, these development-set gains did not transfer to the held-out evaluation data, suggesting that the improvements were confounded by overfitting to the small development set.

2.3 Resources Used

Our system relies on the following publicly available resources: the Qwen3-Embedding-0.6B pretrained model (Team, 2025) served as the backbone encoder; the Sentence-Transformers library (Reimers and Gurevych, 2019) provided the bi-encoder training framework; and PyTorch was used for all custom training loops, including the MLP head and the composite loss implementation. Data augmentation was performed using a T5-based paraphrasing model (Vamsi/T5_Paraphrase_Paws) and synonym replacement to expand the limited training set. No external labeled datasets were used; notably, the 1,900 synthetic training triples provided by the organizers were also not incorporated.

3 System Overview

Our system addresses both tracks through a unified pipeline: a shared bi-encoder backbone produces story embeddings (Track B), and a lightweight MLP head operates on these embeddings for pairwise ranking (Track A). Figure 1 illustrates the complete architecture.

3.1 Track B: Bi-Encoder with Projection Head

The Track B component is a fine-tuned bi-encoder that maps each story to a fixed-size embedding vector. We use Qwen3-Embedding-0.6B (Team, 2025) as the backbone encoder, selected for its strong performance on MTEB benchmarks and its dedicated embedding architecture with last-token pooling. This model was chosen after earlier experiments with BGE-Large-en-v1.5 (Xiao et al., 2023) reached a development-set performance plateau at 0.93 accuracy, suggesting that backbone capacity was the limiting factor.

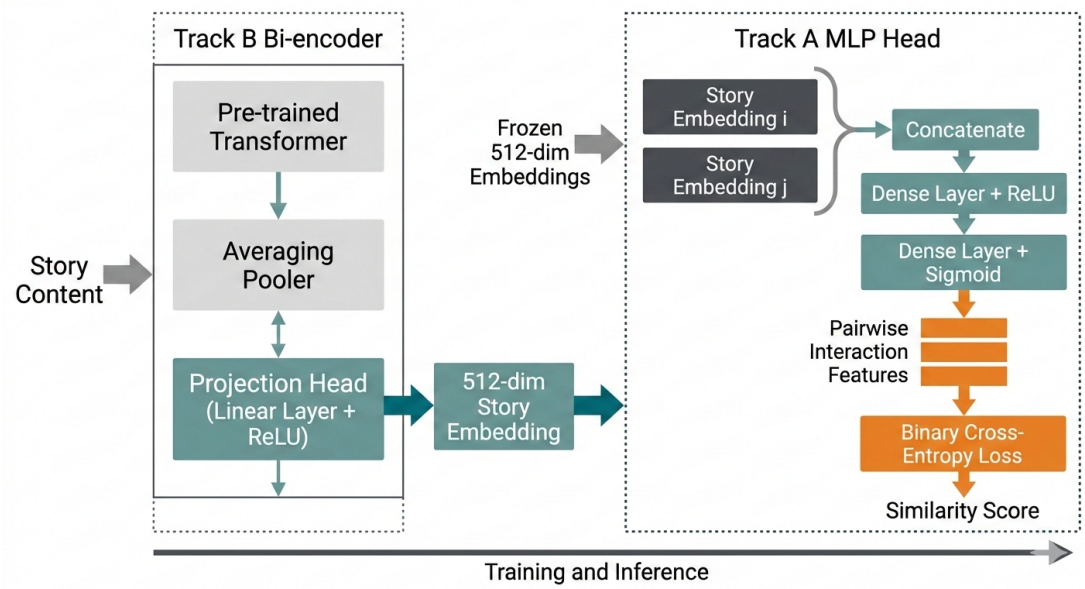

Figure 1: System pipeline. Left: Track B bi-encoder with projection head produces 512-dim story embeddings. Right: Track A MLP head computes pairwise interaction features from the frozen embeddings and outputs a similarity score.

The model is augmented with a linear projection head that reduces the native encoder output from 1024 dimensions to 512 dimensions, followed by L2 normalization:

$$\mathbf{e} = \text{L2Norm}\big(\mathbf{W}_{\text{proj}} \cdot \text{LastTok}(\text{Enc}(\mathbf{x})) + \mathbf{b}\big) \quad (1)$$

where $\mathbf{x}$ is the input story text, $\text{LastTok}(\cdot)$ extracts the last-token hidden state following the Qwen3-Embedding design and official usage examples (Team, 2025), and $\mathbf{W}_{\text{proj}} \in \mathbb{R}^{512\times1024}$ is the projection matrix. The dimensionality reduction to 512 was chosen to improve generalization on the small training set while maintaining representational capacity. $\text{LastTok}(\cdot)$ extracts the last-token hidden state as recommended by the Qwen3-Embedding authors, and $\mathbf{W}_{\text{proj}} \in \mathbb{R}^{512\times1024}$ is the projection matrix. The dimensionality reduction to 512 was chosen to improve generalization on the small training set while maintaining representational capacity.

The bi-encoder is trained with a composite loss combining three objectives:

$$\mathcal{L} = w_m \cdot \mathcal{L}_{\text{margin}} + w_p \cdot \mathcal{L}_{\text{pairwise}} + w_n \cdot \mathcal{L}_{\text{MNR}} \quad (2)$$

where $\mathcal{L}_{\text{margin}}$ is a margin ranking loss (margin = 0.1) that enforces a minimum distance between positive and negative pairs, $\mathcal{L}_{\text{pairwise}}$ is a triple-wise pairwise softmax loss with temperature scaling ($\tau = 0.5$), and $\mathcal{L}_{\text{MNR}}$ is the multiple negatives ranking loss (Henderson et al., 2017) that treats all in-batch non-matching pairs as negatives. The weights $w_m = 1.0$, $w_p = 0.5$, and $w_n = 0.5$ were selected empirically. This simplified composite loss was adopted after experiments showed that adding hard-negative mining, SimCSE-style self-supervision, and curriculum learning to the loss did not improve—and in some cases degraded—development performance. At inference, each story is encoded independently and only once; cosine similarity between embedding pairs is used for evaluation.

Data Augmentation. To mitigate the limited training data, we applied two augmentation strategies: T5-based paraphrasing (generating 2 augmented versions per example) and synonym replacement (replacing up to 3 words per story with probability 0.15). Augmented samples were filtered by cosine similarity to the original (retaining only those in the 0.3–0.9 range) to avoid trivial duplicates or semantically drifted variants, creating 293 new triplets.

3.2 Track A: MLP Head with Pairwise Features

The Track A component is a lightweight MLP head that operates on frozen Track B embeddings to make pairwise ranking decisions. Given a triple (anchor, candidate A, candidate B), the system first encodes all three stories independently using the

frozen bi-encoder. For each anchor–candidate pair, we construct a pairwise feature vector:

$$\phi(\mathbf{e}_a, \mathbf{e}_c) = [\mathbf{e}_a;\ \mathbf{e}_c;\ |\mathbf{e}_a - \mathbf{e}_c|;\ \mathbf{e}_a \odot \mathbf{e}_c] \quad (3)$$

where $\mathbf{e}_a$ and $\mathbf{e}_c$ are the anchor and candidate embeddings, $|\cdot|$ denotes element-wise absolute difference, $\odot$ denotes element-wise product, and $[;]$ denotes concatenation. This four-component representation follows the pairwise interaction features used in natural language inference models (Conneau et al., 2017), yielding a feature vector of dimension $4d = 2048$ for $d = 512$. The feature vector is processed by a two-layer MLP:

$$\text{MLP}(\phi) = \mathbf{W}_2\big(\text{Dropout}(\text{GELU}(\mathbf{W}_1\phi + \mathbf{b}_1))\big) + \mathbf{b}_2 \quad (4)$$

where $\mathbf{W}_1 \in \mathbb{R}^{512\times2048}$, $\mathbf{W}_2 \in \mathbb{R}^{1\times512}$, and the hidden dimension is 512. For each triple, the MLP produces scores $s_A = \text{MLP}(\phi(\mathbf{e}_{\text{anchor}}, \mathbf{e}_A))$ and $s_B = \text{MLP}(\phi(\mathbf{e}_{\text{anchor}}, \mathbf{e}_B))$. The prediction is $\hat{y} = \mathbb{1}[s_A > s_B]$.

The MLP is trained with cross-entropy loss over the stacked scores $[s_A, s_B]$ divided by a temperature $\tau = 0.7$, augmented with a KL divergence distillation term from Track B cosine similarities:

$$\mathcal{L}_A = \mathcal{L}_{\text{CE}} + \lambda \cdot \text{KL}\big(\sigma(\mathbf{s}/\tau_d) \,\|\, \sigma(\mathbf{c}/\tau_d)\big) \quad (5)$$

where $\mathbf{c}$ are the Track B cosine similarity scores acting as teacher, $\tau_d = 0.7$ is the distillation temperature, and $\lambda = 0.3$. This encourages the MLP head to remain consistent with the bi-encoder's similarity judgments while learning to correct its errors.

To ensure robustness on the small dataset, we train 5 independent MLP heads using k-fold cross-validation. At inference, the scores from all heads are averaged before the final prediction:

$$\hat{y} = \mathbb{1}\Big[\frac{1}{K}\sum_{k=1}^{K} s_A^{(k)} > \frac{1}{K}\sum_{k=1}^{K} s_B^{(k)}\Big] \quad (6)$$

3.3 Model Variants Explored

During development, we explored several alternative configurations before converging on the final system. Table 2 summarizes the key variants and their outcomes on the development set. The most significant improvement came from upgrading the backbone from BGE-Large to Qwen3-Embedding-0.6B, while additional training complexity provided marginal or negative returns. However, as described in Section 5, these development-set gains did not generalize to the official test set.

Variant	Dev Acc
BGE-Large backbone	0.86
+ Hard-negative mining	No improvement
+ SimCSE self-supervision	Slight degradation
+ Curriculum learning	No improvement
Qwen3-0.6B backbone	0.94–0.95
+ Simplified composite loss	Best dev result

Table 2: Model variants explored during development. All results are on the development set.

4 Experimental Setup

4.1 Data Splits and Usage

Following the official task setup (Hatzel et al., 2026), the organizers provided two data sources: a labeled development set of 200 triplets and 479 stories for encoding, and a synthetic training set of 1,900 triples generated using large language models. Our system was trained exclusively on the development data; the synthetic training triples were not used. For Track B, the bi-encoder was trained on the full development set with periodic evaluation on Track A triple accuracy as a proxy metric. For Track A, we applied 5-fold stratified cross-validation over the development triples: in each fold, 80% of the triples were used for training and 20% for validation. All 5 resulting MLP heads were retained for ensemble inference. For the final test submission, the models trained on the development set were applied directly to the test data without any additional fine-tuning or retraining.

4.2 Preprocessing

Input stories were tokenized using the Qwen3-Embedding tokenizer with a maximum sequence length of 384 tokens. Stories exceeding this limit were truncated from the right. No additional text cleaning or normalization was applied beyond standard tokenization, as the narrative texts were already well-formed. For data augmentation, T5-based paraphrasing and synonym replacement were applied to expand the training set by approximately 3×, with cosine similarity filtering to discard low-quality augmentations.

4.3 Training Configuration

All experiments were conducted on Google Colab using a single NVIDIA Tesla T4 GPU with 16 GB of VRAM. Training was performed with mixed-precision (fp16) using PyTorch's automatic mixed precision. Gradient checkpointing was enabled to reduce memory consumption during Track B fine-tuning.

Track B. The bi-encoder was fine-tuned using the AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of 2×10^{-5}, weight decay of 0.01 for the backbone and 0.05 for the projection head. We used a batch size of 4 with gradient accumulation over 2 steps (effective batch size of 8), a warmup of 500 steps, and trained for 4 epochs. Gradient clipping was applied with a maximum norm of 1.0.

Track A. Each MLP head was trained using AdamW with a learning rate of 1×10^{-4}, weight decay of 0.01, a warmup ratio of 0.1, and a maximum of 10 epochs with early stopping (patience of 3 epochs, monitored on validation accuracy). The batch size was 8. The temperature for score scaling was set to 0.7 and the distillation weight to 0.3.

Table 3 summarizes the hyperparameter configuration.

Hyperparameter	Track B	Track A
Backbone	Qwen3-0.6B	Frozen
Embedding dim	512	512
Batch size	4 ($\times$2 accum)	8
Learning rate	2e-5	1e-4
Weight decay	0.01 / 0.05	0.01
Epochs	4	10
Temperature	0.5	0.7
Dropout (MLP)	—	0.2
Mixed precision	fp16	fp16
Gradient clip	1.0	1.0
Early stopping	—	patience = 3

Table 3: Hyperparameter configuration for Track B and Track A.

4.4 External Tools and Libraries

The system was implemented in Python 3.11 (Google Colab environment) using PyTorch 2.x for all model training and inference. The Sentence-Transformers library (Reimers and Gurevych, 2019) was used for the bi-encoder training framework. The Hugging Face Transformers library provided the Qwen3-Embedding model and tokenizer. Data augmentation used the T5_Paraphrase_Paws model from the Hugging Face Hub. NumPy was used for embedding I/O (Track B outputs are stored as `.npy` arrays) and scikit-learn for cross-validation splits.

5 Results

5.1 Development Set Results

Table **??** reports the accuracy on the development set. Track B achieves an accuracy of approximately 0.94 using direct cosine similarity between embeddings. Track A, which adds the MLP head ensemble on top of Track B embeddings, improves accuracy to approximately 0.95.

System	Dev Accuracy
Track B (cosine sim)	0.94
Track A (MLP, single fold)	0.95
Track A (MLP, 5-fold ensemble)	0.95

Table 4: Accuracy on the development set.

The k-fold cross-validation results for the Track A MLP head are shown in Table 5. The per-fold variance is notable ($\pm$0.08), which in retrospect was an early warning sign of overfitting that we did not sufficiently heed. Fold 1 achieves perfect accuracy while Folds 3 and 4 drop to 0.80, indicating that the model's strong average performance was unstable.

Fold	Val Accuracy
Fold 0	0.950
Fold 1	0.875
Fold 2	0.950
Fold 3	0.975
Fold 4	0.975
Mean $\pm$ Std	**0.945 $\pm$ 0.0367**

Table 5: K-fold cross-validation results for Track A MLP head.

5.2 Official Test Results

Table 6 reports the official test set results. According to the official shared-task results (Hatzel et al., 2026), the system achieved 53.50% accuracy on Track A (rank 45 out of 47) and 59.25% accuracy on Track B (rank 21 out of 28). The Track A result is near random chance for a binary choice task, representing a catastrophic drop from the 0.95 development accuracy. The Track B result, while

below the median, shows slightly better generalization than Track A, suggesting that the bi-encoder embeddings retained some useful signal despite overfitting.

Track	Test Acc	Dev Acc	Rank
Track A	0.535	0.95	45 / 47
Track B	0.592	0.94	21 / 28

Table 6: Official competition results compared to development accuracy.

The contrast between development and test performance (a drop of over 40 percentage points on Track A) reveals that our system severely overfit to the small development set. We identify two primary causes. First, the system was trained exclusively on the development plus augmented data (493 triples), which is insufficient to learn generalizable narrative similarity patterns; The 1,900 LLM-generated synthetic training triples provided by the organizers were not incorporated into our training pipeline. Incorporating additional synthetic data may improve robustness and mitigate potential distributional shifts between development and test data. Second, the model was applied directly to the test set without any retraining or adaptation, meaning that any distributional shift between development and test data was unmitigated.

5.3 Ablation Study

To assess the contribution of individual components, we conducted an ablation study on the development set. Table 7 reports accuracy when removing or replacing components from the full system. We note that all ablation results are on the development set and should be interpreted with caution given the observed overfitting.

Configuration	Dev Acc
Full system (Track A ensemble)	0.95
− KL distillation ($\lambda = 0$)	0.94
− Ensemble (single fold)	0.95
Track B only (cosine sim)	0.94
− Projection head (1024-dim)	0.93
− Composite loss (MNR only)	0.92
BGE-Large backbone (prev. system)	0.93

Table 7: Ablation study on the development set. These results did not generalize to the test set.

On the development set, the backbone upgrade from BGE-Large to Qwen3-Embedding-0.6B provided the single largest improvement, and the composite loss contributed meaningfully over using MNR alone. However, the test results suggest that these component-level improvements were largely artifacts of overfitting: the gains reflected memorization of development-set patterns rather than genuine improvements in narrative understanding.

5.4 Error Analysis

The near-chance test accuracy on Track A (53.50%) indicates that the system's learned representations do not generalize beyond the development distribution. We identify the following failure modes and contributing factors:

Overfitting to a small training set. The system was trained on approximately 100 labeled triples—far too few for a 0.6B-parameter encoder to learn robust narrative similarity patterns. The high development accuracy (0.95) was misleading, as the model effectively memorized the training examples rather than learning transferable representations. The high per-fold variance in cross-validation (0.80 to 1.00) was an early indicator of this problem.

Unused synthetic training data. The organizers provided 1,900 synthetic triples generated using large language models, which we did not incorporate. This data would have increased the training set by approximately 19×, potentially providing the diversity and volume necessary to learn generalizable patterns. We consider this the single most impactful change we could have made.

No test-time adaptation. The models trained on development data were applied directly to the test set without retraining or fine-tuning. Any distributional shift between the development and test narratives—in terms of genre, style, length, or thematic content—was therefore unmitigated.

Track B vs. Track A generalization. Track B (59.25%) generalized somewhat better than Track A (53.50%), suggesting that the bi-encoder embeddings captured some useful narrative signal despite overfitting, while the MLP head introduced an additional layer of overfitting on top of already overfit embeddings.

6 Conclusion

We presented a two-stage system for SemEval-2026 Task 4: Narrative Similarity, built on Qwen3-Embedding-0.6B. The first stage fine-tunes the en-

coder as a bi-encoder with a 512-dimensional projection head using a composite loss (Track B). The second stage adds a lightweight MLP head trained on frozen embeddings with pairwise interaction features and k-fold ensemble inference (Track A). The system prioritizes parameter efficiency: Each Track A MLP head contains 1.05M parameters; the 5-fold ensemble stores 5.25M parameters in total. The Track B projection head adds 0.525M parameters.

While the architecture achieved strong development accuracy (0.95 on TrackA, 0.94 on TrackB), it generalized poorly to the official test set (53.50% on TrackA, 59.25% on Track B). The primary cause was training on an insufficient amount of data: the system used only the small development set (496 triples) and did not incorporate the 1,900 synthetic training triples provided by the organizers. This experience yields a clear lesson: for narrative similarity on limited data, training set size is more important than architectural sophistication.

Future work will focus on three directions: first, incorporating the synthetic training data and evaluating its impact on generalization; second, applying regularization techniques specifically designed for low-resource fine-tuning, such as R-Drop or mixup at the embedding level; and third, exploring whether larger Qwen3-Embedding variants (4B/8B) can better resist overfitting through stronger pretrained representations.

7 Acknowledgments

The authors would like to acknowledge the support provided by the master's degree scholarship program in engineering at the Universidad Tecnologica de Bolivar (UTB) in Cartagena, Colombia.

References

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017a. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017b. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation.

Nathanael Chambers and Daniel Jurafsky. 2008a. Unsupervised learning of narrative event chains.

Nathanael Chambers and Daniel Jurafsky. 2008b. Unsupervised learning of narrative event chains.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021a. Simcse: Simple contrastive learning of sentence embeddings.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021b. Simcse: Simple contrastive learning of sentence embeddings.

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

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, Laszlo Lukacs, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Marco Marelli, Stefano Menini, Marco Baroni, Raffaella Bernardi, and Roberto Zamparelli. 2014. A sick cure for the evaluation of compositional distributional semantic models.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. Mteb: Massive text embedding benchmark.

Andrew J. Reagan, Lewis Mitchell, Dilan Kiley, Christopher M. Danforth, and Peter Sheridan Dodds. 2016. The emotional arcs of stories are dominated by six basic shapes.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Qwen Team. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-pack: Packaged resources to advance general chinese embedding.