Scaling Sentence Similarity for Classical Tibetan with Automatic Annotations

Shay Cohen[1,2], Jingyi Yang[3], Gal Rabinovitz[1,2], Sonam Choden[3], Ofir Shtrosberg[1,2], Nicola Bajetta[3], Goody Ben Horin[1,2], Rebecca Sundén[3], Omri Drori[1,2], Sonam Jamtsho[3], Dorji Wangchuk[3], Kfir Bar[1,2], Orna Almogi[3], Shai Fine[1]

[1]Data Science Institute, Reichman University, Herzliya, Israel
[2]Efi Arazi School of Computer Science, Reichman University, Herzliya, Israel
[3]University of Hamburg, Germany

Correspondence: shay.cohen02@post.runi.ac.il

Abstract

Identifying intertextual parallels is central to philology, traditionally requiring labor-intensive manual analysis. While digitized historical corpora enable automated approaches using semantic sentence embeddings, training such models requires large annotated datasets, which are scarce for low-resource languages. We address this challenge by introducing a scalable automatic annotation pipeline for training semantic embedding models for Classical Tibetan. Our method combines unsupervised contrastive bootstrapping with iterative pair mining, generating silver-standard similarity labels through two complementary annotation strategies: (1) an ensemble of embedding models and rerankers, and (2) an LLM-as-a-judge committee using best–worst scaling. When combined with a domain-specific, gold-standard annotated dataset for sequential fine-tuning, the resulting text-similarity model achieves a state-of-the-art Spearman correlation of 0.864 on the STS task. This enables effective semantic search in Classical Tibetan and provides a framework for automatic supervision in low-resource languages used in digital humanities.

1 Introduction

The digitization of global textual heritage has unlocked vast new opportunities for computational analysis within the Digital Humanities. A prime example is the Tibetan Buddhist literary corpus, the focus of this study, which represents one of the largest and most complex collections of pre-modern literature in the world. Spanning philosophy, logic and epistemology, ritual and meditation, traditional sciences, language and literature, and the arts. This textual archive, now being preserved by organizations such as the Buddhist Digital Resource Center (BDRC[1]), offers a profound window into human intellectual history. However, its sheer scale and the linguistic complexities of Classical Tibetan make it largely inaccessible to modern, large-scale computational methods.

A central scholarly interest in working with such corpora is intertextuality: the reuse, adaptation, and transformation of earlier materials across texts, genres, intellectual circles, and historical periods. Tibetan literature, in particular, is deeply interwoven through quotation, borrowing, commenting, and doctrinal reformulation. Tracing these relationships is essential for understanding intellectual transmission and historical development. However, rather than mere verbatim repetition, intertextual connections often involve paraphrase, stylistic harmonization, condensation, or expansion, making simple string matching insufficient. To detect intertextuality at this non-trivial semantic level, we require methods capable of identifying meaning-preserving variation across passages. A foundational tool for this task is a sentence similarity model. Such a model would enable scholars to perform semantic search, identify textual reuse, trace the evolution of concepts, and cluster related passages—tasks that are currently laborious or impossible. Yet, the development of high-performance models, such as sentence transformers (Reimers and Gurevych, 2019), is fundamentally dependent on massive, human-labeled datasets.

For Classical Tibetan, such resources are virtually nonexistent, making this requirement a critical bottleneck. The number of experts worldwide with the linguistic and doctrinal depth needed to annotate semantic similarity is exceptionally small. As a result, creating a gold-standard dataset through traditional annotation is logistically impractical and

[1]https://www.bdrc.io

Proceedings of the 6th International Conference on Natural Language Processing for the Digital Humanities, pages 150–166

highly labor-intensive.

To address this data-scarcity challenge, we introduce and evaluate a novel pipeline for training a text-similarity model for Classical Tibetan using synthetic data, achieving performance comparable to models trained on gold-standard annotations. By creating an iterative pipeline that mines candidate pairs from an unannotated corpus and synthetically annotates them using model ensembles, we construct a large-scale, cost-effective, and high-quality training dataset. We use this data, alongside expert-annotated examples, to train `Tib-Bi-Dharma`, a dedicated sentence transformer for Classical Tibetan. Our contributions are **threefold**:

1. We release `Tib-Bi-Dharma`, a new foundational text-similarity embedding model that unlocks a high-dimensional semantic search and new possibilities for computational study of the vast Tibetan literary heritage.
2. We release comprehensive Classical Tibetan sentence-pair annotation datasets, comprising both a large-scale synthetically scored corpus and a novel expert-annotated gold-standard dataset, providing a crucial benchmark for future text-similarity research.
3. We introduce an end-to-end pipeline that dynamically mines candidate text pairs and automatically annotates them, serving as a methodological framework for overcoming data scarcity in low-resource languages.

2 Related Work

The paradigm for large-scale semantic similarity was established by Sentence-BERT (SBERT) (Reimers and Gurevych, 2019), which maps sentences independently to dense vector spaces to bypass the quadratic computational bottleneck of standard cross-encoder architectures (Devlin et al., 2019). Subsequent work, such as SimCSE (Gao et al., 2021), further improved embedding quality by introducing advanced contrastive learning objectives, notably by reformulating Natural Language Inference (NLI) datasets into discrete triplets. However, because these discrete classification objectives push pairs either together or apart, they can introduce a discrepancy between the training loss and continuous evaluation metrics. To resolve this, CoSENT (Huang et al., 2024) utilizes a ranking-based loss to handle continuous similarity labels, ensuring the embedding space accurately reflects fine-grained semantic differences.

While highly effective, these methods fundamentally rely on massive, human-labeled training datasets. For low-resource domains like Classical Tibetan, this requirement is often insurmountable. A common alternative is the use of large-scale multilingual models, such as mBERT (Devlin et al., 2019) or XLM-R (Conneau et al., 2020). Although these models exhibit impressive cross-lingual transfer, their representation spaces are diluted across over 100 languages, causing them to underperform compared to dedicated monolingual models.

To address general data scarcity, Thakur et al. (2021) proposed Augmented SBERT, utilizing a cross-encoder for labeling and Kernel Density Estimation (KDE) for pair sampling. We follow the spirit of this approach in our zero-resource setting by substituting the standard cross-encoder with either a multi-model representation ensemble or a generative LLM-as-a-judge committee for automated data labeling (Section 3.2.3). Furthermore, we replace static KDE with an iterative, active semantic mining loop for dynamic pair sampling (Section 3.2.2).

In recent years, significant progress has been made in Tibetan NLP (Huang et al., 2025; Meelen et al., 2024). Foundational models, most notably the monolingual TiBERT (Liu et al., 2022) and Tibetan-BERT-wwm (Liang et al., 2024), have been effective for tasks like text classification and sentence boundary disambiguation. Furthermore, multilingual models like CINO (Yang et al., 2022) have shown strong performance across the broader Sino-Tibetan language family.

Despite this progress, a critical research gap remains: the lack of a dedicated, high-performance sentence-transformer model optimized for Classical Tibetan. While Modern and Classical Tibetan share the same script, the archaic vocabulary and the distinct morphology of historical texts render modern-trained models suboptimal. Furthermore, while models like TiBERT offer strong general-purpose language capabilities, they are not calibrated for sentence-level semantic similarity out-of-the-box. Consequently, applications like semantic search or clustering, even for Modern Tibetan (Engels and Barnett, 2025), currently must rely on unoptimized multilingual baselines. This paper addresses this gap with `Tib-Bi-Dharma`. By integrating state-of-the-art contrastive bootstrapping, synthetic data generation, and continuous scaling objectives, our approach provides a holistic, empirically validated solution for Classical Tibetan

sentence embeddings.

3 Methodology

Training a high-quality text similarity model requires a large number of semantically aligned text pairs. In the context of historical corpora, such pairs typically take the form of textual parallels: passages that convey equivalent or closely related meanings, even when expressed through paraphrase, condensation, expansion, or stylistic variation. Unlike surface-level duplication, these parallels reflect deeper semantic correspondence and are essential for modeling meaning-preserving variation. For a sentence transformer to learn a meaningful embedding space, it must observe both positive examples of such semantic equivalence and contrasting negative examples that differ in meaning. Constructing this type of parallel data at scale is therefore central to effective similarity learning.

However, in Classical Tibetan, such parallel data is extremely scarce and costly to annotate manually. To address this limitation, we adopt a two-tiered strategy. First, we curate a high-quality "gold-standard" dataset with human experts, primarily serving as a robust evaluation benchmark. Second, to address the scalability limitations of manual annotation, we propose and evaluate a pipeline for automatically annotating text parallels at scale, and use the resulting data to train and evaluate text-similarity models.

Although Classical Tibetan uses its own script, many academic and computational resources adopt the Wylie transliteration scheme (Wylie, 1959), which represents Tibetan characters with the Latin alphabet. We use Wylie for all experiments unless otherwise stated.

3.1 Gold-Standard Dataset of Text Parallels

Seven scholars of Tibetan Studies compiled the set, which comprises hand-picked pairs drawn from known Buddhist corpora, including the *bKa' 'gyur*, *bsTan 'gyur*, *rNying ma rgyud 'bum*, as well as *Rong zom gsung 'bum*. To ensure semantic diversity and strict traceability, each candidate pair picked by the team was pre-categorized into four similarity levels and recorded on a corresponding spreadsheet. To maintain rigorous data provenance, we explicitly logged both the identity of the scholar who originally sourced each candidate pair and the annotator who provided the final evaluation. Once there were enough pairs in each category pool, we selected 1,000 pairs with a balanced class distribution to facilitate unbiased relative scoring.

These 1,000 pairs were distributed among the seven scholars and annotated in four discrete batches: an initial warm-up batch of 100 pairs, followed by three subsequent batches of 300 pairs each. To mitigate the subjective biases inherent in absolute rating scales, we employed Best-Worst Scaling (BWS) (Louviere and Woodworth, 1991; Louviere et al., 2015), implemented via an established BWS framework (Saif M. Mohammad). During the task, annotators were presented with randomly sampled 4-tuples of Classical Tibetan sentence pairs and instructed to isolate exactly one "most similar" (best) and one "least similar" (worst) pair per tuple (see Table 4 in Appendix A for examples). To maintain strict annotation quality, any highly ambiguous or challenging comparisons were escalated to a senior annotator for guidance and final adjudication. To convert these discrete categorical selections into a continuous similarity score S_i for each sentence pair i, we applied the standard BWS scoring formula: $S_i = \frac{B_i - W_i}{N_i}$ where B_i and W_i represent the total number of times pair i was selected as the most and least similar, respectively, and N_i denotes the total number of times the pair appeared across all evaluated tuples. This rigorous comparative approach, combined with senior oversight, allowed us to compute highly reliable continuous similarity scores for the final publicly released gold-standard dataset[2]. Table 5 (Appendix B) provides examples of these final scored pairs.

3.2 Text-Similarity Model Creation

We describe a process for building a Classical Tibetan text-similarity model, in which candidate training pairs are iteratively retrieved from the raw corpus and annotated through active learning.

3.2.1 Corpus Preprocessing and Segment Pool Generation

Our raw corpus consists of the Kangyur and Tangyur collections[3]. In order to generate text segments from the continuous, space-less Classical Tibetan script, we apply a custom segmentation algorithm that produces segments of 10–30 words. We use the first two million segments (out

[2] https://huggingface.co/datasets/Intellexus/tibetan-sts-gold

[3] Sourced from https://github.com/Esukhia/derge-kangyur and https://github.com/Esukhia/derge-tengyur

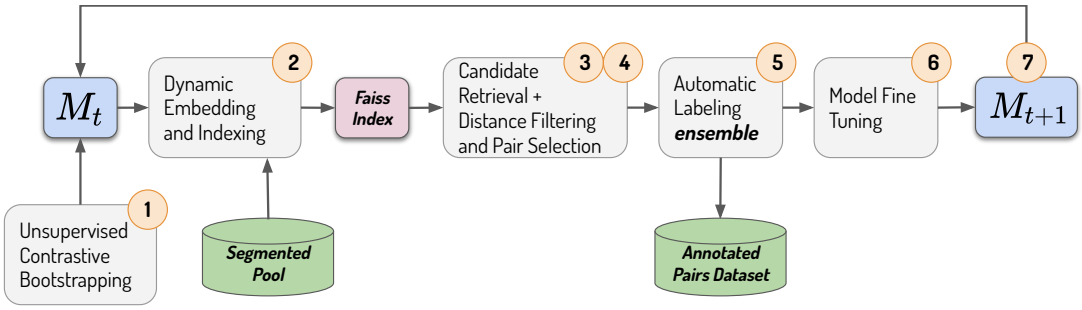


Figure 1: End-to-end active learning pipeline for candidate pair mining. M_t represent the model at the i'th iteration.

of 2,823,001) as our primary training pool. Full preprocessing and segmentation details are provided in Appendices H and I.

3.2.2 Automatic Annotation of Segment Pairs

Training a text-similarity model in a contrastive setting requires not only semantically aligned positive pairs, but also informative hard negatives to provide a robust learning signal (Xiong et al., 2021). Our preliminary experiments show that without explicit embedding-oriented fine-tuning, a foundation model exhibits severe anisotropy, consistent with prior findings on representation degeneration (Ethayarajh, 2019; Gao et al., 2019). As reported by Li et al. (2020), text-segment vectors collapse into a narrow cone in the embedding space, and we observe the same phenomenon: cosine similarities from the uncalibrated model are uniformly high regardless of true semantic overlap.

Our objective is to extract semantically related pairs from this vast, unannotated pool, allowing the model to learn an embedding space in which meaning-preserving variations yield high cosine similarities. To overcome this initial anisotropy and accomplish our objective, we designed a pipeline (illustrated in Figure 1) that actively mines candidate segment pairs based on the model's evolving similarity estimates.

1. **Unsupervised Contrastive Bootstrapping:** We start with a base model. As a preliminary step to mitigate anisotropy, we bootstrap the model by fine-tuning it with unsupervised contrastive learning over the entire pool of segments. Specifically, we train the model for three epochs by treating the same segment as anchor-positive alignment pairs and optimizing with the Multiple Negatives Ranking Loss, where the negatives consist of the other examples in the same batch, following the SimCSE framework. This effectively forces the collapsed vector space to expand and disperse prior to our mining phase.
2. **Dynamic Embedding and Indexing:** The current iteration of the sentence embedding model encodes all remaining, unused textual segments in the training pool into dense vector representations. The newly generated embeddings are then ingested and indexed using `Faiss` (Douze et al., 2024) to facilitate rapid, high-dimensional similarity operations.
3. **Candidate Retrieval:** To optimize computational efficiency, we randomly sample a batch of 250 segments from the training pool simultaneously, which we use as queries. Using their dense vector representations, we perform a batched query against the `Faiss` index to retrieve the top-k ($k = 1,000$) most semantically similar candidate segments based on cosine distance.
4. **Distance Filtering and Pair Selection:** For each query, we randomly select a target segment from its top-k neighbors, enforcing a strict cosine distance threshold of ≥ 0.6 to guarantee the extraction of semi-hard candidates with high information gain. Selected pairs are permanently removed from the pool to prevent redundancy. If filtering yields fewer than 250 valid pairs, we iteratively repeat Step 3 until the target batch size is reached or the pool is exhausted. In the unlikely event that no valid pairs can be extracted, the pipeline halts and requires manual recalibration of the initial threshold settings.
5. **Automatic Labeling:** Extracted pairs are automatically labeled using a strategy selected from Section 3.2.3 and kept fixed for the entire pipeline run. We experiment with two strategies and report their performance.
6. **Model Fine-Tuning:** The sentence embedding model is contrastively trained using the CoSENT loss on this newly acquired batch of

automatically annotated pairs, effectively updating its weights and refining its spatial alignment.

7. **Iteration:** The loop restarts at Step 2. The newly updated model generates fresh embeddings for the remaining pool, ensuring that the next batch of mined pairs is selected based on an incrementally improved vector space.

To facilitate future research, we publicly release the code for our pipeline framework[4]. Using this framework across multiple iterations, we generated automatically annotated pairs and aggregated them into a final, comprehensive dataset to train our downstream text-similarity models.

3.2.3 Automatic Annotation of Text Parallels

In Step 5, the extracted segment pairs are automatically annotated. Here, we explore and evaluate two different automated annotation approaches. The first leverages the capabilities of existing text-similarity models (text embeddings and re-rankers), while the second leverages generative large language models (LLMs) instructed to emulate the human expert annotation process using BWS.

Approach 1: Sentence Embeddings and Re-ranking Models (EMR) In our first approach, we evaluated a range of pre-trained sentence embedding models and encoder–decoder re-ranking models (Guo et al., 2016) to assess their ability to capture semantic similarity in parallel texts. Sentence embedding models encode each sentence independently into a fixed-dimensional vector space, with similarity computed via cosine similarity, whereas encoder–decoder re-rankers jointly process both sentences as a concatenated input and directly predict a similarity score. To leverage complementary strengths, we construct a late-fusion ensemble in which the final similarity score is computed as a weighted average of the top-performing models' outputs.

We begin with a zero-shot evaluation of several state-of-the-art embedding models and encoder–decoder re-rankers on our manually curated gold-standard dataset of Classical Tibetan sentence pairs. Performance is measured using standard correlation metrics, Spearman and Pearson, to assess alignment between model predictions and human similarity judgments. Full results are provided in Table 6 (Appendix C). Based on this evaluation, we select the three strongest models: BGE[5] (Chen et al., 2024b; Xiao et al., 2024), GemmaEmbedding-300m[6] (Schechter Vera et al., 2025), and Qwen3-Reranker-8B[7] (Zhang et al., 2025). The first two are embedding-based models, while the third is a re-ranking architecture. For the ensemble, we experiment with different weight combinations (listed in Table 7 in Appendix D) and select a balanced configuration that achieves nearly the highest Spearman correlation. Table 1 summarizes the performance of the three individual models and their combined ensemble.

Model	Spearman (ρ)	Pearson (r)
BGE	0.809	0.806
Qwen3-Reranker-8B	0.809	0.753
GemmaEmbedding-300m	0.785	0.782
Ensemble	**0.839**	0.827

Table 1: Zero-shot evaluation of text-similarity models.

Approach 2: Generative LLMs Using BWS As an alternative to conventional embedding-based similarity, we leverage chat LLMs by framing them as a committee of expert annotators, an approach that has shown strong effectiveness in automating complex annotation tasks (Bagdon et al., 2024; Zhang and Feng, 2025). We employ the same BWS protocol used in our human expert annotation process (Section 3.1). To construct the ensemble, we adopt an early-fusion strategy: instead of computing separate BWS scores per model and averaging them, we merge the raw categorical selections ("best" and "worst" for each 4-tuple) produced by the LLMs into a single annotation pool. The BWS standardization procedure is then applied to this combined set of judgments, effectively simulating a multi-annotator setting.

To identify effective models for this task, we benchmark a diverse set of state-of-the-art generative LLMs using the BWS protocol against our gold-standard dataset (full results in Table 8, Appendix E). To keep the pipeline financially scalable, we focus on lightweight, cost-effective variants. Our final committee therefore includes two high-performing instruction-tuned models from different vendors: `Gemini 3 Flash (preview)` (Google DeepMind, 2026a) and `Claude 4.5 Haiku` (Anthropic, 2025). Table 2 reports the correlation of

[4] https://github.com/Intellexus-DSI/sensim

[5] BAAI/bge-multilingual-gemma2

[6] google/embeddinggemma-300m

[7] Qwen/Qwen3-Reranker-8B

each model, and their ensemble, with human expert annotations on the gold-standard dataset.

Model	Spearman (ρ)	Pearson (r)
Gemini 3 Flash	0.843	0.837
Claude 4.5 Haiku	0.811	0.809
Ensemble	**0.847**	0.844

Table 2: Evaluation results of LLMs using BWS.

As demonstrated in both Table 1 and Table 2, both ensembling techniques consistently outperform their respective individual models.

3.2.4 Training Text-Similarity Models

Using the automatically annotated data produced by the pipeline described above, we train two encoder-based language models: BERT and ModernBERT. Prior to contrastive learning for text similarity, both models undergo continual pretraining using the standard masked-language modeling (MLM) loss on a large corpus of Wylie-transliterated Classical Tibetan to better capture the language's lexical and syntactic properties. The resulting models are referred to as Tib-B (BERT-based) and Tib-MB (ModernBERT-based). Additional details are provided in Appendix J.

4 Experiments and Results

4.1 Experimental Setup

We first run our annotation pipeline for 20 iterations for each supervision type to generate two collections of labeled segment pairs: one using EMR supervision (3.2.3) and another using LLM-based BWS supervision (3.2.3). In Step 1 of the annotation pipeline, we use Tib-B as the foundation model to select candidate pairs from the training pool. For the BWS approach, we employ a $4N$ 4-tuple strategy (N denoting the total number of sentence pairs), intentionally exceeding the standard $1.5N$–$2N$ baseline as Kiritchenko and Mohammad (2017) show that further oversampling improves annotation quality.

Before training Tib-B and Tib-MB on the automatically annotated pairs, we perform a preliminary training step to adapt the model for text similarity. We experiment with two approaches for this initialization step: (1) **Anchor–Positive (AP):** Following Gao et al. (2021), identical segments with dropout noise are used to form anchor–positive pairs. We use this standard setup as a reference baseline for the bootstrapped model described in Step 1 of the annotation pipeline. (2) **Consecutive Segments (CS):** To capture narrative continuity and discourse flow, we construct positive pairs by coupling adjacent text segments from the unannotated corpus (i.e., treating segment t_{i+1} as the positive target for t_i). This approach follows self-supervised methods that exploit document-level context by treating neighboring sentences as semantically related (Logeswaran and Lee, 2018; Giorgi et al., 2021).

Both initialization approaches are optimized using the multiple-negatives-ranking loss (Henderson et al., 2017), which combats initial representation degeneration by pulling positive pairs together and treating all other in-batch segments as hard negatives.

Finally, the models are fine-tuned for seven epochs using the CoSENT objective (Huang et al., 2024), chosen because it naturally handles our continuous scores while maintaining robust contrastive alignment. Training utilized the AdamW optimizer (learning rate 2e-5, temperature $\lambda = 20$) on NVIDIA RTX 3090/GB10 GPUs (see Appendix K for the full hyperparameter configuration).

4.2 Evaluation Metrics

Following standard practices for Semantic Textual Similarity (STS) tasks (Reimers and Gurevych, 2019), we compute the cosine similarity between sentence embeddings. Model performance is then evaluated using Spearman's rank correlation coefficient (ρ) between these predicted similarities and the human judgments in our novel Classical Tibetan gold dataset (Section 3). Additional details regarding the STS setup are provided in Appendix F. To ensure robust evaluation, all experiments employ 4-fold cross-validation to generate the training, validation, and test splits. Appendix G provides further details and the data distributions in Table 9.

4.3 Comparison with Tibetan Foundation Models

We compare our models to several state-of-the-art Tibetan language models: LaBSE (Feng et al., 2022), TiBERT (Liu et al., 2022), CINO (Yang et al., 2022), and T-RoBERTa[8] . Since these models are trained on native Unicode Tibetan script, we convert our data from Wylie transliteration to Unicode before evaluation, allowing them to fully utilize their Tibetan tokenizers and vocabularies.

[8] https://huggingface.co/sangjeedondrub/tibetan-roberta-base

Model	Baseline	Gold	EMR	BWS	EMR →Gold	BWS →Gold	EMR+BWS →Gold
LaBSE	0.803±.02	0.831±.03	0.830±.02	0.840±.03	0.836±.02	0.845±.02	0.844±.02
CINO	0.615±.01	0.766±.02	0.771±.01	0.763±.02	0.793±.02	0.788±.02	0.805±.02
T-RoBERTa	0.778±.03	0.798±.03	0.804±.03	0.795±.04	0.806±.03	0.804±.04	0.810±.03
TiBERT	0.691±.03	0.747±.02	0.755±.04	0.738±.03	0.770±.03	0.773±.04	0.781±.04
Tib-B	0.795±.02	0.846±.01	0.839±.02	0.843±.02	0.853±.02	0.855±.02	0.856±.02
Tib-B-AP	0.807±.02	0.833±.01	0.827±.02	0.838±.02	0.844±.02	0.842±.02	0.853±.01
Tib-B-CS	0.801±.01	0.844±.01	**0.845**±.01	0.847±.02	0.849±.01	0.850±.02	0.852±.02
Tib-MB	0.702±.02	0.832±.02	0.824±.02	0.827±.03	0.842±.02	0.842±.03	0.848±.02
Tib-MB-AP	0.689±.04	0.754±.04	0.797±.02	0.792±.03	0.811±.02	0.815±.02	0.823±.01
Tib-MB-CS	**0.808**±.01	**0.851**±.01	0.841±.02	**0.850**±.02	**0.859**±.01	**0.864**±.02	**0.858**±.01

Table 3: Spearman (ρ) correlation results for all evaluated models. **AP** denotes initialization using anchor-positive pairs with multiple-negatives-ranking loss, while **CS** denotes initialization using consecutive segments. The model Tib-B-AP was used as the unsupervised bootstrap model in the automatic annotation pipeline.

4.4 Data Configurations

We train the models with contrastive learning on different data sources and combinations: (1) **Baseline:** no training; (2) **Gold**[9]**:** training on the gold dataset only; (3) **EMR**[10]**:** EMR-annotated pairs; (4) **BWS**[11]**:** LLM-based BWS-annotated pairs; (5) **EMR→Gold:** EMR followed by Gold; (6) **BWS→Gold:** BWS followed by Gold; and (7) **EMR +BWS→Gold:** EMR+BWS followed by Gold.

4.5 Results and Discussion

Unsupervised Bootstrapping Efficacy. As shown in Table 3, unsupervised initialization substantially improves baseline performance. The CS strategy consistently outperforms AP, raising the Tib-MB Spearman correlation from 0.702 to 0.808 and indicating that adjacent segments provide a stronger signal of semantic continuity.

The Advantage of Prior Sentence Alignment. As a multilingual sentence embedder, LaBSE shows a strong zero-shot advantage, reaching a baseline Spearman correlation of 0.803. Under the BWS→Gold regime, it achieves 0.845, indicating that cross-lingual sentence spaces adapt well to Classical Tibetan.

Automatic vs. Gold Training. Training only on the BWS dataset yields performance comparable to the human-annotated Gold set. For instance, Tib-MB-CS scores 0.850 on BWS versus 0.851 on Gold, while LaBSE and Tib-B-CS perform slightly better on BWS. This suggests that the scale and diversity of BWS effectively compensate for the absence of expert annotations.

Effective Knowledge Transfer from the Ensemble A notable result from the standalone automatic training experiments is the effectiveness of knowledge transfer. Trained only on EMR or BWS data, and without any Gold labels, our lightweight Tib-MB-CS model achieves Spearman correlations of 0.841 and 0.850, surpassing the zero-shot performance of the larger ensemble used to generate the labels.

Two-Stage Training Yields Best Performance Table 3 shows that two-stage training consistently outperforms single-dataset training. The best configuration, BWS→Gold, first trains on the large automatic pool and then refines on Gold, yielding a peak Spearman correlation of 0.864 for Tib-MB-CS. We release Tib-Bi-Dharma[12] with this top configuration, demonstrating that a lightweight model trained with automatic and gold data can outperform large multilingual ensembles.

4.6 Qualitative Analysis of the Automatic Annotation Pipeline

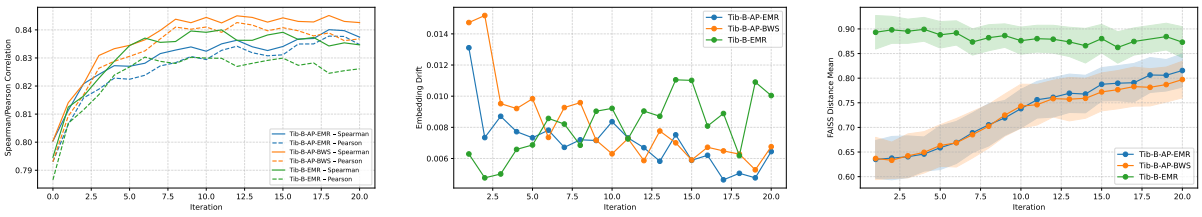
To analyze how the embedding space evolves across iterations in the annotation pipeline, we track three metrics over 20 training iterations. Figure 2 shows the trajectories of downstream test performance (Spearman and Pearson correlations), embedding-space drift of the first 10^5 segments in the training pool, and candidate retrieval dynamics for both EMR and BWS supervisions. We explain and

[9] https://huggingface.co/datasets/Intellexus/tibetan-sts-gold

[10] https://huggingface.co/datasets/Intellexus/tibetan-sts-synthetic-emr

[11] https://huggingface.co/datasets/Intellexus/tibetan-sts-synthetic-bws

[12] https://huggingface.co/Intellexus/Tib-Bi-Dharma-Tibetan-EWTS-v2



(a) Downstream STS Correlations (b) Embedding Drift (c) Mean `Faiss` Retrieval Distance

Figure 2: STS performance correlation (a), embedding drift (b), and mean `Faiss` retrieval distance (c) across 20 iterations of the annotation pipeline for different models.

analyze each metric below. To measure the effect of the AP initialization on the `Tib-B` model, we compare the initialized and uninitialized versions across all three metrics. For brevity, the uninitialized baseline shown is the model trained with `EMR`.

Performance Trajectory and Convergence Figure 2a shows Spearman and Pearson correlations on the test set across iterations. Performance improves rapidly in the early stages, indicating that the first batches of automatically annotated pairs effectively establish the basic semantic structure. After iterations 10–15, the metrics plateau, suggesting the model has largely captured the available signal and that additional pairs yield diminishing returns.

Embedding Drift Analysis We track embedding drift (Figure 2b), defined as the average cosine distance between a segment's embedding at iteration t and $t-1$. For the initialized models (`Tib-B-AP-EMR` and `Tib-B-AP-BWS`), drift is highest in the early iterations, reflecting rapid reorganization after unsupervised bootstrapping, and gradually stabilizes as the pipeline adds semantic supervision. In contrast, the uninitialized model (`Tib-B-EMR`) starts from a collapsed anisotropic space and requires more iterations to separate the representations. By iteration 20, the initialized models reach a stable lower drift, indicating that the learning loop has largely stabilized the embedding space.

Retrieval Dynamics and Candidate Hardness Figure 2c tracks the mean and standard deviation of `Faiss` cosine distances for retrieved candidate pairs, where the mean distance reflects candidate hardness. In the initialized models, the initial dispersion yields high retrieval distances that gradually decrease as the model learns to bring semantically related segments closer together. In contrast, the uninitialized model (`Tib-B-EMR`) begins in a collapsed anisotropic space with artificially low distances and exhibits the opposite trend, as the iterative process progressively separates the representations. Despite these different starting points, the bootstrapped models consistently operate within the 0.60–0.80 range, an effective regime for mining informative training pairs (Xiong et al., 2021).

5 Conclusion and Future Work

In this work, we introduced `Tib-Bi-Dharma`, significantly improving semantic similarity modeling for Classical Tibetan and enabling reliable semantic search across Tibetan Buddhist literature. By validating a fully automated annotation framework, we demonstrate that the long-standing expert annotation bottleneck can be mitigated through automatic supervision: training solely on automatically annotated data matches gold-standard performance, and combining both yields a peak correlation of 0.864. Our approach allows scholars to trace conceptual development and textual reuse across centuries of texts. Beyond Classical Tibetan, the methodology provides a general blueprint for building high-quality retrieval systems in other low-resource cultural heritage domains. Our work opens several directions for future research. First, we plan to move from sentence-level alignment to corpus-scale discovery, using `Tib-Bi-Dharma` to detect intertextual parallels across the entirety of Tibetan Buddhist literature (including both allochthonous and autochthonous). This will enable large-scale analysis of conceptual transmission and textual reuse. We also aim to extend the embedding space cross-lingually by linking Tibetan passages to their Sanskrit sources and applying the pipeline to other low-resource textual traditions. Finally, we will refine the annotation pipeline with improved acquisition strategies that target uncertain or contested sentence pairs, further reducing annotation effort.

Limitations

This work has several limitations. First, the automatic annotation pipeline introduces noise into the training data, as automatically labeled parallels may include semantically weak or incorrect pairs despite our ranking and BWS-based scoring strategies. Second, the method relies on the availability of large unannotated corpora from which candidate pairs can be extracted, an assumption that may not hold for languages with extremely limited textual resources. Third, because candidate parallels are extracted from the same corpus, the learned representations may partly reflect corpus-specific stylistic or topical patterns rather than general semantic equivalence.

Acknowledgments

This study is supported in part by the European Research Council (Intellexus, Project No. 101118558). Views and opinions expressed are, however, those of the authors only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authorities can be held responsible for them.

References

Anthropic. 2025. Claude 4.5 haiku (version 20251001). Large Language Model, API model string: `claude-haiku-4-5-20251001`. Accessed: 2026-02-22 to 2026-03-03.

Christopher Bagdon, Prathamesh Karmalkar, Harsha Gurulingappa, and Roman Klinger. 2024. "you are an expert annotator": Automatic best–worst-scaling annotations for emotion intensity modeling. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7924–7936, Mexico City, Mexico. Association for Computational Linguistics.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024a. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *Preprint*, arXiv:2402.03216.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024b. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2318–2335, Bangkok, Thailand. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

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.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library.

James Engels and Robert Barnett. 2025. Developing a semantic search engine for modern tibetan. *Revue d'Etudes Tibétaines*, 74:262–283.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. 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 55–65, Hong Kong, China. Association for Computational Linguistics.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891, Dublin, Ireland. Association for Computational Linguistics.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tieyan Liu. 2019. Representation degeneration problem in training natural language generation models. In *International Conference on Learning Representations*.

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.

John Giorgi, Osvald Nitski, Bo Wang, and Gary Bader. 2021. DeCLUTR: Deep contrastive learning for unsupervised textual representations. In *Proceedings of the 59th Annual Meeting of the Association for*

Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 879–895, Online. Association for Computational Linguistics.

Google DeepMind. 2025a. Gemini 2.5 flash. Large Language Model, API model string: `gemini-2.5-flash`. Accessed: 2026-02-22 to 2026-03-03.

Google DeepMind. 2025b. Gemini 2.5 flash-lite. Large Language Model, API model string: `gemini-2.5-flash-lite`. Accessed: 2026-02-22 to 2026-03-03.

Google DeepMind. 2026a. Gemini 3 flash (preview). Large Language Model, API model string: `gemini-3-flash-preview`. Accessed: 2026-02-22 to 2026-03-03.

Google DeepMind. 2026b. Gemini 3.1 pro (preview). Large Language Model, API model string: `gemini-3.1-pro-preview`. Accessed: 2026-02-22 to 2026-03-03.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM'16, page 55–64. ACM.

Kai Golan Hashiloni, Shay Cohen, Asaf Shina, Jingyi Yang, Orr Meir Zwebner, Nicola Bajetta, Guy Bilitski, Rebecca Sundén, Guy Maduel, Ryan Conlon, Ari Barzilai, Daniel Mass, Shanshan Jia, Aviv Naaman, Sonam Choden, Sonam Jamtsho, Yadi Qu, Harunaga Isaacson, Dorji Wangchuk, and 3 others. 2025. DharmaBench: Evaluating language models on buddhist texts in Sanskrit and Tibetan. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pages 2088–2110, Mumbai, India. The Asian Federation of Natural Language Processing and The 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. *Preprint*, arXiv:1705.00652.

Xinshuo Hu, Zifei Shan, Xinping Zhao, Zetian Sun, Zhenyu Liu, Dongfang Li, Shaolin Ye, Xinyuan Wei, Qian Chen, Baotian Hu, Haofen Wang, Jun Yu, and Min Zhang. 2025. Kalm-embedding: Superior training data brings a stronger embedding model. *Preprint*, arXiv:2501.01028.

Cheng Huang, Nyima Tashi, Fan Gao, Yutong Liu, Jiahao Li, Hao Tian, Siyang Jiang, Thupten Tsering, Ban Ma-bao, Renzeg Duojie, Gadeng Luosang, Rinchen Dongrub, Dorje Tashi, Jin Zhang, Xiao Feng, Hao Wang, Jie Tang, Guojie Tang, Xiangxiang Wang, and 3 others. 2025. Tibetan language and ai: A comprehensive survey of resources, methods and challenges. *Preprint*, arXiv:2510.19144.

Xiang Huang, Hao Peng, Dongcheng Zou, Zhiwei Liu, Jianxin Li, Kay Liu, Jia Wu, Jianlin Su, and Philip S. Yu. 2024. Cosent: Consistent sentence embedding via similarity ranking. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:2800–2813.

Svetlana Kiritchenko and Saif Mohammad. 2017. Best-worst scaling more reliable than rating scales: A case study on sentiment intensity annotation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 465–470, Vancouver, Canada. Association for Computational Linguistics.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130, Online. Association for Computational Linguistics.

Chaofan Li, Zheng Liu, Shitao Xiao, and Yingxia Shao. 2023. Making large language models a better foundation for dense retrieval. *Preprint*, arXiv:2312.15503.

Yatao Liang, Hui Lv, Yan Li, La Duo, Chuanyi Liu, and Qingguo Zhou. 2024. Tibetan-bert-wwm: A tibetan pretrained model with whole word masking for text classification. *IEEE Transactions on Computational Social Systems*, 11(5):6268–6277.

Sisi Liu, Junjie Deng, Yuan Sun, and Xiaobing Zhao. 2022. Tibert: Tibetan pre-trained language model. In *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2956–2961.

Lajanugen Logeswaran and Honglak Lee. 2018. An efficient framework for learning sentence representations. In *International Conference on Learning Representations*.

Jordan J Louviere, Terry N Flynn, and Anthony Alfred John Marley. 2015. *Best-worst scaling: Theory, methods and applications*. Cambridge University Press.

Jordan J Louviere and George G Woodworth. 1991. Best-worst scaling: A model for the largest difference judgments. Technical report, working paper.

Marc Marone, Orion Weller, William Fleshman, Eugene Yang, Dawn Lawrie, and Benjamin Van Durme. 2025. mmbert: A modern multilingual encoder with annealed language learning. *Preprint*, arXiv:2509.06888.

Marieke Meelen, Sebastian Nehrdich, and Kurt Keutzer. 2024. Breakthroughs in tibetan nlp; digital humanities. *Revue d'Etudes Tibétaines*.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. 2022. Sentence-t5: Scalable sentence encoders from pretrained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*,

pages 1864–1874, Dublin, Ireland. Association for Computational Linguistics.

OpenAI. 2025. Gpt-5 mini. Large Language Model, API model string: `gpt-5-mini-2025-08-07`. Accessed: 2026-02-22 to 2026-03-03.

OpenPecha. a. Botok: a powerful python library for tokenizing tibetan text. GitHub repository.

OpenPecha. b. pyewts: Python tibetan unicode to wylie (ewts) converter. GitHub repository.

Nils Reimers, Philip Beyer, and Iryna Gurevych. 2016. Task-oriented intrinsic evaluation of semantic textual similarity. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 87–96, Osaka, Japan. The COLING 2016 Organizing Committee.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Saif M. Mohammad. Best-worst-scaling-scripts: Code to assist with best-worst-scaling annotations.

Henrique Schechter Vera, Sahil Dua, Biao Zhang, Daniel Salz, and 1 others. 2025. Embeddinggemma: Powerful and lightweight text representations. *arXiv preprint arXiv:2509.20354*.

Octen Team. 2025. Octen series: Optimizing embedding models to 1 on rteb leaderboard.

Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. 2021. Augmented SBERT: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 296–310, Online. Association for Computational Linguistics.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. 2024. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. *Preprint*, arXiv:2412.13663.

Turrell V Wylie. 1959. A standard system of tibetan transcription. *Harvard Journal of Asiatic Studies*, 22:261–267.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2024. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '24, page 641–649, New York, NY, USA. Association for Computing Machinery.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*.

Ziqing Yang, Zihang Xu, Yiming Cui, Baoxin Wang, Min Lin, Dayong Wu, and Zhigang Chen. 2022. CINO: A Chinese minority pre-trained language model. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3937–3949, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Peng Yu, En Xu, Bin Chen, Haibiao Chen, and Yinfei Xu. 2025. Qzhou-embedding technical report. *Preprint*, arXiv:2508.21632.

Mengchen Zhang and Xiang Feng. 2025. Automated annotation of academic emotion intensity in online learning comment texts: A bws method based on llms. In *Proceedings of the 2024 16th International Conference on Education Technology and Computers*, ICETC '24, page 317–323, New York, NY, USA. Association for Computing Machinery.

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. *arXiv preprint arXiv:2506.05176*.

Xinping Zhao, Xinshuo Hu, Zifei Shan, Shouzheng Huang, Yao Zhou, Xin Zhang, Zetian Sun, Zhenyu Liu, Dongfang Li, Xinyuan Wei, Youcheng Pan, Yang Xiang, Meishan Zhang, Haofen Wang, Jun Yu, Baotian Hu, and Min Zhang. 2025. Kalm-embedding-v2: Superior training techniques and data inspire a versatile embedding model. *Preprint*, arXiv:2506.20923.

A Gold BWS 4-tuple pairs example

This table presents five examples of 4-tuples used during the annotation process. For each tuple of four sentence pairs, the annotator (whether human or synthetic) must select the "most similar" (best) and "least similar" (worst) pair.

Ex.	Pair	Sentence A	Sentence B
1	1	slob ma ni dad pa che zhing sgrub pa la dga' ba/ gsang sngags la mos shing dbang don du gnyer pa/	slob ma dad ldan sgrub la dga'// rtsa ba'i sngags bzlos byed pa dang // gsang sngags rgyud la dga' sems shing // gzhan yang phyi ma'i don 'dod pa'o//
	2	sgyu ma'i skyes bus sgyu ma'i skyes bu dag la chos bstan pa bzhin no/	sems rgyun yan lag can dang mtshungs//
	3	de la ni chos rdul tsam yang mi dmigs so//	de ni nam mkha'i mtshan nyid do//
	4	de ni bcad par gyur pa yin zhes bstan// gang du che ba bzhi yis btab nas su// sbrul gyis ni zlog par gyur te// bstan nas bsad pa yin zhes bshad pa ste//	de ni bcad par gyur pa yin zhes bstan// gang du mche ba bzhi yis btab nas su// sbrul gyi lus ni zos par gyur pa de// btab nas zos pa yin zhes bshad pa ste//
2	1	nam mkha'i mtshan nyid snga rol na// nam mkha' cung zad yod ma yin// gal te mtshan las snga gyur na// mtshan nyid med par thal bar 'gyur// mtshan nyid med pa'i dngos po ni// 'ga' yang gang na'ang yod ma yin//	bdag dang bdag gir 'dzin byas pa// 'dus byas spyod yul can chags ni// rgyu yin de yi gnod byed ni// bdag med mthong ba 'gal ba yin//
	2	de'i dngos po'i ngo bo nyid rnam par brtags na rdul phra mo tsam gyi chos rnams kyang dmigs su med do//	gang tshe 'dus byas 'dus ma byas dang dkar nag chos// shes rab rnam par bshig nas rdul tsam mi dmigs tshe// 'jig rten dag na shes rab pha rol phyin grangs 'gro// nam mkha' gang la'ang cung zad mi gnas de dang 'dra//
	3	de bzhin du rig byed smra ba la sogs pa ltar shes pa po ni bdag yin no snyam du rtog cing / rnam par shes pa'i ngo bo nyid du bzung ste/	bdag tu 'dzin pa'i rnam shes ni// bdag las gal te skye 'gyur na// des na kun bdag nyid rnam shes//
	4	sems ni bza' btung tshogs la chags mi bya//	bza' btung sogs la sems ni chags mi bya//
3	1	sems ni sgyu ma'i rnam pa ste// byang chub kyang ni sgyu ma 'dra//	mya ngan las 'das pa ni mya ngan las 'das pas stong ngo//
	2	de ltar sangs rgyas bsam mi khyab// sangs rgyas chos kyang de bzhin te// bsam mi khyab la dad pa yi// rnam par smin pa'ang de bzhin no//	sangs rgyas bcom ldan yon tan bsam mi khyab// dam pa'i chos kyi yon tan bsam mi khyab// 'phags pa'i dge 'dun yon tan bsam mi khyab// dkon mchog bsam mi khyab la mngon dad pa'i// rnam smin bla na med pa'ang bsam mi khyab//
	3	de bzhin byang chub sems ni ma bskyed cing // sangs rgyas ma dad lha gzhan la brten pas// nga yi gsang sngags bzlas na phung bar 'gyur//	de dag nga yi dkyil 'khor gzhug / drag po g.yo dang mi g.yo ba// drag tu 'khrul cing sems can rnams// nga yi dkyil 'khor nang du gzhug /de mthong sems can 'chi bar 'gyur//
	4	chos thams cad ni sems tsam ste/ sems dang mtshungs par ldan pa yin no//	sems nyid kyis ni sems can 'jig rten dang // snod kyi 'jig rten shin tu sna tshogs 'god// 'gro ba ma lus las las skyes par gsungs// sems spangs nas ni las kyang yod ma yin//
4	1	sring mo de ltar na byang chub dang mig gi rnam par shes pa'i khams de ni gnyis su med de gnyis su dbyer med do//	sring mo de ltar na byang chub dang mig gi rnam shes kyi khams de ni gnyis su med de gnyis su dbyer med do//
	2	dkyil 'khor chen po bshad du gsol//	gzugs brnyan dkyil 'khor bri bar bya//
	3	yan lag bcu gnyis yod ma yin// mtha' dang mtha' med pa yang med// lta ba thams cad spang ba'i phyir// sems tsam du ni ngas bshad do//	gang gis thugs brtse nyer bzung nas// lta ba thams cad spang ba'i phyir// dam pa'i chos ni ston mdzad pa// gau tam de la phyag 'tshal lo//
	4	dka' thub nges par mi bzad pas// bsten pas 'grub par mi gyur gyi//	dka' thub sdom pa mi bzad pa// brten pas 'grub par mi 'gyur te//
5	1	dod chags dang / zhe sdang dang / gti mug gang yin pa ste/ de dag ni/ spang bar bya ba yin pas yongs su spangs pa la rab tu brtson zhing bslab par bya'o//	gti mug chags dang zhe sdang dang / nga rgyal phrag dog mi spang do/
	2	nyon mongs de dag gang gi yin// de yang grub pa yod ma yin// gal te gang med ci zhig yod// nyon mongs cung zad yod ma yin//	nyon mongs de dag gang gi yin// de yang grub pa yod ma yin// 'ga' med par ni gang gi yang // nyon mongs pa dag yod ma yin//
	3	sdang ba'i dgra bo'i srog rtsa gcod par mdzod//	dam nyams dgra bo'i srog rtsa gcod mdzad pa'i//
	4	dbang ma thob pas mi rig par// sgrub pos 'khor lo de bya'o//	sangs rgyas sprin rnams de yi tshe// de la dbang bskur rab tu rtsol//

Table 4: Examples of 4-tuples evaluated during annotation.

B Gold BWS scored pairs example

This table provides five examples of sentence pairs along with their normalized BWS similarity scores.

Ex.	Sentence A	Sentence B	Score
1	de lta bas na lta ba mtho dman ni snang ba la dngos por zhen pa che chung gi bye brag tsam ste/	lta ba'i bye brag 'di dag kyang rdzas su yod med kyi gzhi las brtsams te/ lta ba'i dbye ba mdor bsdus te bstan pa'o//	0.75
2	byang chub tu sems bskyed pas khros pas gzhan dag 'tshogs pas sdom pa 'chor bar 'dus/	byang chub sems dpar sems bskyed nas dam chos 'drar snang ston pas spong /	0.5625
3	ci'i phyir 'byung ba'i bdag nyid lhan cig skyes pa'i ye shes su gyur// lhan cig skyes pa nyid kyis de brjod ces bya ba'i don to//	de dag rang gis ngo ma shes pa ni/ lhan cig skyes pa'i ma rig pa ste/ ma rig pa de 'khor ba'i gzhir rgyur gyur pa las 'byung khungs kyi sgra/ zhes bya'o//	0.375
4	dag pa dang ba 'od gsal ba// mi 'khrugs 'dus ma byas pa ni// byang chub sems dpa'i spyod yul lo//	sgyu ma'i rjes su mthun pa ni shin tu dang ba'i me long gi gzugs brnyan lta bu'o//	0.1875
5	gal te bdag dang sdug pa dang // rtag dang bde ba yod na ni// bdag shes sdug shes rtag shes dang // bde shes phyin ci log ma yin//	gal te bdag dang gtsang ba dang // rtag dang bde ba yod na ni// bdag dang gtsang dang rtag pa dang // bde ba phyin ci log ma yin//	0.875

Table 5: Sentence pairs alongside their computed normalized BWS scores.

C Sentence Embeddings and Re-ranking Models Evaluation

Table 6: Single-Model Evaluation Results against gold standard

Type	Model	Spearman (ρ)	Pearson (r)
embedding	`BAAI/bge-multilingual-gemma2` (Chen et al., 2024b; Xiao et al., 2024)	0.809	0.806
reranker	`Qwen/Qwen3-Reranker-8B` (Zhang et al., 2025)	0.809	0.753
embedding	`Octen/Octen-Embedding-8B` (Team, 2025)	0.792	0.800
embedding	`tencent/KaLM-Embedding-Gemma3-12B-2511` (Zhao et al., 2025; Hu et al., 2025)	0.792	0.789
embedding	`BAAI/bge-m3` (Chen et al., 2024b)	0.790	0.792
reranker	`BAAI/bge-reranker-v2-gemma` (Li et al., 2023; Chen et al., 2024a)	0.786	0.773
embedding	`Qwen/Qwen3-Embedding-8B` (Zhang et al., 2025)	0.786	0.796
embedding	`google/embeddinggemma-300m`[13]	0.785	0.782
embedding	`sentence-transformers/sentence-t5-xxl` (Ni et al., 2022)	0.754	0.738
embedding	`Kingsoft-LLM/QZhou-Embedding` (Yu et al., 2025)	0.716	0.701

D Sentence Embeddings and Re-ranking Models Ensemble Evaluation

Table 7: Multi-Model Ensemble Evaluation Results against the gold standard. The ensemble highlighted in **bold** was selected as our annotation ensemble for embedding models and rerankers.

Multi-Model Ensembles	Weights	Spearman (ρ)	Pearson (r)
`bge-multilingual-gemma2`	0.40	0.840	0.831
`embeddinggemma-300m`	0.20		
`bge-m3`	0.10		
`Qwen3-Reranker-8B`	0.30		
`bge-multilingual-gemma2`	0.33	**0.839**	0.827
`embeddinggemma-300m`	0.33		
`Qwen3-Reranker-8B`	0.34		
`bge-multilingual-gemma2`	0.30	0.839	0.822
`embeddinggemma-300m`	0.30		
`Qwen3-Reranker-8B`	0.40		
`bge-multilingual-gemma2`	0.30	0.839	0.822
`embeddinggemma-300m`	0.10		
`bge-m3`	0.20		
`Qwen3-Reranker-8B`	0.40		
`bge-multilingual-gemma2`	0.25	0.838	0.832
`embeddinggemma-300m`	0.25		
`bge-m3`	0.25		
`Qwen3-Reranker-8B`	0.25		
`bge-multilingual-gemma2`	0.33	0.830	0.800
`embeddinggemma-300m`	0.33		
`Qwen3-Reranker-4B`	0.34		
`bge-multilingual-gemma2`	0.25	0.828	0.802
`embeddinggemma-300m`	0.25		
`bge-m3`	0.25		
`Qwen3-Reranker-0.6B`	0.25		
`embeddinggemma-300m`	0.25	0.796	0.755
`bge-m3`	0.25		
`Qwen3-Reranker-0.6B`	0.25		
`bge-reranker-v2-gemma`	0.25		

E Generative LLMs Using BWS Evaluation

Guided by Hashiloni et al. (2025), Table 8 details our evaluation of the following generative LLMs:

- `Gemini 3 Flash` (Google DeepMind, 2026a)
- `Gemini 3.1 Pro` (Google DeepMind, 2026b)
- `Gemini 2.5 Flash` (Google DeepMind, 2025a)
- `Gemini 2.5 Flash-Lite` (Google DeepMind, 2025b)
- `Claude 4.5 Haiku` (Anthropic, 2025)
- `GPT-5 Mini` (OpenAI, 2025)

Table 8: Generative LLMs Using BWS evaluation against gold standard

Model	Spearman (ρ)	Pearson (r)
`Gemini 3 Flash`	0.843	0.837
`Gemini 3.1 Pro`	0.836	0.832
`Gemini 2.5 Flash-Lite`	0.796	0.794
`Gemini 2.5 Flash`	0.821	0.818
`Claude 4.5 Haiku`	0.811	0.809
`GPT-5 Mini`	0.794	0.789

F Adapting Annotations for Semantic Textual Similarity (STS)

To evaluate our automatically annotated datasets on the downstream STS task, the `EMR` and `BWS` scores must be mapped to a standard Semantic Textual Similarity (STS) scale.

Data Formulation A standard STS dataset consists of sequence pairs and a similarity score. We formulate our annotated dataset as a collection of tuples, $D = \{(x_i, y_i, s_i)\}$, where x_i and y_i represent the i-th pair of text segments (Sentence A and Sentence B), and the label s_i represents the similarity score.

Data Normalization While `EMR` produces continuous absolute similarity scores in the range $s_i^{\text{EMR}} \in [0, 1]$, `BWS` yields continuous relative scores in the range $s_i^{\text{BWS}} \in [-1, 1]$. We therefore normalize the `BWS` scores to a unified scale of $s_i \in [0, 1]$ to ensure comparability across annotation schemes prior to any usage. This normalization is applied to both the gold-standard and synthetic-standard datasets.

Bi-Encoder Inference During both evaluation and training, the language model operates as a strict bi-encoder. Let f_θ denote the sentence transformer model parameterized by θ. For a given pair, (x_i, y_i), the model independently encodes both segments, using a pooling method (as specified in Appendix K), and outputs dense vector representations: $\mathbf{u}_i = f_\theta(x_i)$ and $\mathbf{v}_i = f_\theta(y_i)$. The predicted semantic similarity between the two segments is then computed using cosine similarity: $\hat{s}_i = \cos(\mathbf{u}_i, \mathbf{v}_i)$.

Evaluation Metric The primary evaluation metric, as suggested by Reimers et al. (2016), is Spearman's rank correlation (ρ). To evaluate model performance on the held-out test set, we compute Spearman's rank correlation coefficient (ρ) between the gold-standard scores and the model predictions.

G Gold and Synthetic Sets Distribution

To ensure robust evaluation, the gold dataset was partitioned into four cascading folds with strictly non-overlapping test sets. To completely isolate the validation data across iterations, the validation set for each successive fold was explicitly sampled as a subset of the preceding fold's test set. This cascading structure guarantees that the validation sets are completely disjoint from one another and never overlap with the current test set.

Table 9: Annotated sets

Set	Size
Train	600
Validation	150
Test	250
Gold	1,000
BWS (Synthetic)	5,000
EMR (Synthetic)	5,000

H Corpus Preprocessing and Segmentation Pipeline

The text preprocessing pipeline is designed to transform continuous Classical Tibetan text—specifically in this study, the Kangyur and Tengyur corpora 3—into dynamically sized context windows that respect natural linguistic boundaries optimized for the embedding model. This is achieved through a sequential, three-step process: cleaning the raw input, parsing and aggregating it into semantically coherent spans bounded by strict length constraints, and finally converting those spans into the Extended Wylie Transliteration Scheme (EWTS).

H.1 Text Cleaning

The pipeline begins by standardizing the raw input text. Using regular expressions, the text is filtered

to retain only characters within the Tibetan Unicode block. All non-Tibetan characters, extraneous non-Tibetan punctuation, and Latin scripts are removed, and whitespace is standardized to create a clean baseline for grammatical analysis. Additionally, if the downstream architecture requires strictly contiguous character strings, the pipeline can be configured to completely strip all remaining intra-segment whitespace.

H.2 Segmentation and Span Generation

Once cleaned, the Unicode text is parsed into foundational grammatical "atoms." For this study, we utilize Botok (OpenPecha, a), a high-accuracy token-based engine that analyzes lexical tokens to reliably resolve the syntactic ambiguity of the Tibetan *shad* (Wylie / or |) punctuation mark (detailed in Appendix I). To prevent excessive semantic fragmentation during this atomic phase, we enforce a strict length constraint: any otherwise valid syntactic split is deterministically suppressed if the preceding sequence contains fewer than four syllables.

Immediately following this atomic segmentation, these base Unicode units are dynamically concatenated into discrete, non-overlapping text spans. The pipeline sequentially aggregates adjacent atoms until the cumulative length satisfies specific upper and lower bounds, which in our experiments were configured to target a window of 10 to 30 words. This guarantees that each resulting sequence provides sufficient semantic context without exceeding the embedding model's optimal input capacity.

H.3 Transliteration

In the final step, the correctly sized Tibetan Unicode spans are passed through a deterministic conversion module. This generates the corresponding EWTS strings, safely transliterating the constrained text segments into Latin characters to ensure compatibility with the downstream foundation models.

I Guidelines for Segmenting a Tibetan Corpus

I.1 General Guidelines

Double or quadruple *shad* (Wylie / or |) or *gter tsheg* (Wylie :) should be treated as strong boundaries, as in prose they mark the end of long syntactic units (such as passages, paragraphs, or chapters) or the closing of specific short syntactic units (such as citations or expressions of thoughts), and in verse the end of a line. Segmentation should generally be implemented at these occurrences.

Single *shad* or *gter tsheg* should be treated as weak boundaries, as they mark the end of short syntactic units, such as clauses, phrases, or sentences. Segmentation at these occurrences depends on specific conditions:

a. Split after a single *shad* when it is immediately preceded by:

- Sentence terminators (final particles): *go; ngo; do; no; bo; mo; 'o; ro; lo; so; to;*
- Optative/imperative suffixes, which commonly immediately follow an adjective, a verb, an adverbialized adjective, or an adverbialized verb: *cig; gyur cig; zhig; shig; shog; rogs; rogs gnang.*

b. Split after a single *shad* when it is immediately followed by:

- Section markers/enumerations: *dang po; gnyis pa;* etc.;
- Sentence-initial markers: *de nas; de bas na; de'i rjes; de lta bas na; de yang; der yang; de ma yin; de ma thag; de ma gtogs; de min; de phyir; de'i phyir; de bzhin du; des na; gzhan du na; gzhan yang; yang na; 'o na; 'on te; 'on kyang; gal te; gal srid; slar yang; spyir la; spyir btang; spyir yang; mdor na; mdor bsdus na.*

c. Do NOT split after a single *shad* when it is immediately preceded by:

- Gerund markers: *nas; bzhin; bzhin du; bzhin pa; bzhin par; kyin;*
- Ablative, locative, genitive, conjunctive, and instrumental particles: *la; su; du; na; ru; tu; nas; las; gi; gyi; kyi; -'i; 'i; yi; phyir; dang; zhing; cing; ste; te; kyang; yang; -'ang; 'ang; -'am; 'am; pas; bas; ltar; gis; -n/m/r/l gyis* (excluding *par gyis, bar gyis*); *kyis; yis;*
- Adverbializer markers: *par; bar;*
- Topic marker: *ni.*

d. Do NOT split after a single *shad* when a correlative pair (e.g., *ji ltar* <> *de ltar; ji srid* <> *de srid; ji snyed* <> *de snyed; ji tsam* <> *de tsam*) is present in the text but not complete at the point of the *shad*. In other words, when in the chunk preceding the *shad* any of the following are present: *ci zhig; ci ste; ci 'dra; ci ltar; ci bzhin; ci tsam; ji snyed; ji srid; ji ltar; ji bzhin; ji skad; ji tsam,*

refrain from splitting before any of the following correlative forms appear (if present): *de bzhin; de ltar; de tsam; de skad; de 'dra.*

In the case of citations, split immediately before the chunk containing the opening phrase (e.g., *de ltar kyang bkod pa chen po las//*) and immediately after the chunk containing the closing phrase (e.g., *zhes gsungs pa lta bu'o//*). However, the rules of c. and d. still apply and take priority. For example, in the following case: *da ni dkyil 'khor bri ba'i thabs ston te/ de ci'i phyir zhe na/ gru bzhi pa la sgo bzhi pa// sgo ba rnams dang ldan pa ste// rta babs bzhi dang ldan par bya// lte ba dang ni mu khyud ldan// zhes gsungs te/ gru bzhi ni phrin las rnam pa bzhis 'gro ba'i don byed pa'i rtags yin pas gru bzhir ldan no//,* instead of splitting immediately before *de ci'i phyir zhe na/* and immediately after *zhes gsungs te/*, treat the entire sequence as a single unit.

In the case of *mantra*s and *dhāraṇī*s (i.e., a string of syllables that do not follow the rules for Tibetan syllables), treat them as one separate unit, ignoring all *shad*s in between.

I.2 Clarification on Certain Conventions and Their Transliteration

The rules described above concerning the application of a double *shad* are not always followed by Tibetan scribes. Some apply a single *shad* systematically also after a final particle or at the end of a verse line. In such cases, a single *shad* should be regarded as equivalent to a double *shad*.

A space in the Tibetan script, often following any type of *shad*, is rendered in Extended Wylie with an underscore.

Any space in texts transliterated in Wylie following any type of *shad* (i.e., a space representing an actual space in the Tibetan script) should be distinguished from those typically corresponding to the syllable separator *tsheg* and should therefore be treated as an underscore.

A *shad* is often omitted when it follows the letter *ga*, which can create ambiguity in transliteration. Therefore if in the original Tibetan text the final word of a syntactic unit ends in the letter *ga* (not *nga*) without a *shad*, and the end of the unit is indicated only by a space, it should be interpreted as terminating with a single *shad*. Likewise, if the final word ends with the letter *ga* (not *nga*) with a single *shad*, it should be interpreted as terminating with a double *shad*.

In this way, sometimes the complication is taken into consideration in transliterated texts, so that letter *ga* plus space in Tibetan script is transliterated as *-g/; ga/; gi/; gu/; ge/; go/*, each marked as being followed by an underscore, and letter *ga* plus a single *shad* plus space is transliterated as *-g//; ga//; gi//; gu//; ge//; go//*, each marked as being followed by an underscore. However, that is not always the case, in which scenario only the application of a double *shad* can be deduced with certainty (i.e., after a final particle or at the end of a verse line).

If such spaces as described above are not represented by underscores in the transliteration and are instead rendered as ordinary spaces, cases of omitted *shad* after letter *ga* become difficult to identify.

J Tibetan models

J.1 Data Collection and Processing

We compiled a corpus of e-texts of Classical Tibetan canonical, para-canonical, and non-canonical scriptures and non-scriptures totaling 13.8 GB of clean running text. The data sources include publicly available repositories such as the Spither Tibetan corpus, as well as in-house collections curated by our team of scholars. We also incorporated corpora that may contain limited amounts of modern Tibetan, such as the MC2 corpus. To ensure data quality, we applied document-level deduplication using the MinHash algorithm combined with a Locality-Sensitive Hashing (LSH) index for efficient similarity lookups. Documents with an estimated Jaccard similarity of 0.8 were considered duplicates and removed. Subsequently, we performed rule-based cleaning and normalization to remove comments, HTML artifacts, and long numerical sequences. We also applied paragraph-level filtering, discarding paragraphs with fewer than four tokens, excessively repetitive word patterns, or consecutive duplicates—issues that often result from OCR errors in digitized manuscripts.

J.2 Transliteration Conversion

Some of the data were converted between Unicode and Wylie using a tool we developed which mainly leverages pyewts (OpenPecha, b). The tool operates through a three-stage cascade: (1) a Unicode heuristic that identifies the dominant script family using Unicode code-point categories; (2) a statistical profile classifier that applies extended langdetect n-gram models trained on Sanskrit and Tibetan transliteration profiles; and (3) a deterministic fallback that uses curated regular-expression templates for Sanskrit schemes and structural stack patterns

for Tibetan, validated through pyewts round-trip checks. It supports Sanskrit (Devanāgarī, IAST, SLP-1, Harvard–Kyoto, Velthuis) and Tibetan (Unicode, Wylie/EWTS, ACIP), with additional coverage for Modern Chinese (CJK) and English.

J.3 Tib-B (BERT-based)

We trained a Tibetan-adapted variant of mBERT by continually pre-training bert-base-multilingual-cased (Devlin et al., 2019) on the full 13.8 GB corpus described above. The tokenizer was trained from scratch using the WordPiece model with a target vocabulary size of 32,000 and the standard special tokens ([UNK], [CLS], [SEP], [PAD], [MASK]), optimized for Tibetan. This yields a 30.8 percent reduction in token count compared to the original mBERT tokenizer (fertility of 1.43 vs. 2.07). Continual pre-training followed standard masked language modeling (MLM) with a masking probability of 0.15, a batch size of 8 per device, sequence length of 512, learning rate of 2e-5, AdamW optimizer with weight decay of 0.01, warmup ratio of 0.06, FP16 mixed precision, gradient clipping at a max norm of 1.0, and 3 training epochs.

J.4 Tib-MB (ModernBERT-based)

Tib-MB builds upon jhu-clsp/mmBERT-base (Marone et al., 2025), a multilingual BERT variant based on the ModernBERT (Warner et al., 2024) architecture. The base model comprises 311,871,168 parameters distributed across 22 hidden layers with 12 attention heads, a hidden size of 768, and an intermediate (FFN) size of 1,152. It uses RoPE positional embeddings (both local and global = 160,000) with a maximum positional embedding capacity of 8,192 tokens, and global attention applied every 3 layers. Before training, the tokenizer was extended to support Tibetan EWTS transliteration system. A SentencePiece BPE model was trained on the same 13.8 GB Tibetan corpus with a target vocabulary of 8,000 subword tokens and a character coverage of 0.9995. After filtering tokens already present in the original vocabulary, 5,389 new tokens were added, expanding the final vocabulary from approximately 255,923 to 261,312 tokens. The model's embedding matrix was resized accordingly, with newly added embeddings initialized randomly. The model was continually pre-trained via masked language modeling (MLM) with a masking probability of 0.15. Input documents were tokenized and split using a sliding window strategy: chunk size of 1,024 tokens with a 512-token overlap (stride = 512), yielding multiple overlapping training examples per long document. Chunks shorter than 100 tokens were discarded. Training was conducted for 3 epochs with a per-device batch size of 1 and 8 gradient accumulation steps (effective batch size of 8), learning rate of 2e-5 with linear decay scheduler and 100 warmup steps, AdamW optimizer (= 0.9, = 0.999, = 1e-8) with weight decay of 0.01, and gradient clipping at a max norm of 1.0. Training was performed in FP16 mixed precision with gradient checkpointing enabled to accommodate the longer sequences on a single NVIDIA RTX 4000 Ada Generation GPU (12 GB VRAM).

K Experimental Training Hyperparameters

To ensure the reproducibility of our experiments, Table 10 details the comprehensive set of hyperparameters utilized during the training in our experiments, including the `Tib-Bi-Dharma` model.

Table 10: Models Hyperparameter configurations

Hyperparameter	Value
Epochs	7
Optimizer	AdamW (fused)
Learning Rate	2e-5
Learning Rate Scheduler	Reduce LR on Plateau
Pooling	Weighted Mean
Loss Function	CoSent
Loss Scale	20.0
Warmup Steps	64
Weight Decay	0.1
Max Gradient Norm	1.0
Batch Size (per device)	256
Gradient Accumulation Steps	1
Precision	bf16
Seed	42