Comparative Study of Word2Vec, FastText and Multilingual BERT for Uzbek Natural Language Processing

Abdilatif Meyliev[1], Farrukh Ishkobilov[1], Mironshoh Ortikov[1], Leyla Cumayeva[2] and Javlon Ibragimov[1]

[1]*Department of Information Technologies, University of Information Technologies and Management, Kurgan Str. 1, 180212 Karshi, Uzbekistan*

[2]*Department of Foreign Languages, Azerbaijan University of Architecture and Construction, AZ1073 Baku, Azerbaijan*

abdulatif1909@gmail.com, farruxishqobilov@gmail.com, mironshoh5500@gmail.com, bqodirovv@icloud.com, leyla.jumayeva@azmiu.edu.az

Keywords: Uzbek Language, Word Embeddings, Word2Vec, FastText, Multilingual BERT, Low-Resource NLP, Contextual Embeddings.

Abstract: Natural language processing research has achieved remarkable progress with the development of word embedding techniques; however, most existing studies focus on high-resource languages. Low-resource and morphologically rich languages such as Uzbek still face significant challenges related to data sparsity and vocabulary variability. This paper presents a comparative analysis of static, subword-aware, and contextual embedding models for the Uzbek language within a unified and reproducible experimental framework. Specifically, Word2Vec, FastText, and multilingual BERT (mBERT) are evaluated using intrinsic evaluation metrics, including vocabulary size, out-of-vocabulary (OOV) coverage, and average cosine similarity. The experiments are conducted on a preprocessed Uzbek text corpus that combines educational, scientific, and synthetically generated data to mitigate resource limitations. The results demonstrate that Word2Vec struggles with morphological variation and OOV words, while FastText significantly improves vocabulary coverage through subword modeling. Furthermore, mBERT produces rich contextual sentence-level representations, capturing semantic information beyond the capabilities of static embeddings. The findings highlight the importance of subword-aware and contextual embedding approaches for low-resource languages and provide practical insights into embedding selection for Uzbek natural language processing tasks. The proposed framework provides a reproducible baseline for future research on embedding models and downstream NLP applications for the Uzbek language. All experiments and results are generated using an open-source codebase, ensuring full transparency and reproducibility through a publicly available repository.

1 INTRODUCTION

Natural Language Processing (NLP) has become one of the most influential fields of artificial intelligence, enabling machines to understand, analyze, and generate human language. In recent years, significant progress has been achieved through the development of word embedding techniques, which represent textual units as dense numerical vectors capturing semantic and syntactic relationships [1], [2].

Despite these advances, the majority of NLP research and resources remain concentrated on high-resource languages such as English, Chinese, and German. Low-resource languages, including Uzbek, face substantial challenges due to limited annotated corpora, scarce linguistic resources, and complex morphological structures. In such languages, a large number of word forms emerge from a single root [3] due to extensive inflection and derivation, leading to severe data sparsity and out-of-vocabulary issues.

Classical embedding models such as Word2Vec have demonstrated strong performance in capturing semantic similarity; however, they suffer from limitations in handling out-of-vocabulary (OOV) words and morphological variations [1]. To address these issues, subword-based approaches such as FastText were introduced, allowing embeddings to be generated from character n-grams [4]. More recently, contextual embedding models based on Transformer architectures, such as multilingual BERT (mBERT), have shown remarkable results by producing context-aware representations at the sentence level [5].

This study aims to conduct a comprehensive comparative analysis of static, subword-aware, and contextual embedding models for the Uzbek language. Specifically, Word2Vec, FastText, and mBERT are evaluated within a unified experimental framework. The research focuses on intrinsic evaluation metrics, including vocabulary coverage, OOV handling capability, and semantic similarity, to assess the suitability of each approach for low-resource and morphologically rich language scenarios.

The main contributions of this work can be summarized as follows:

1) A reproducible experimental pipeline for training and evaluating Uzbek embedding models is proposed.
2) A comparative analysis of Word2Vec, FastText, and mBERT is conducted on an Uzbek text corpus.
3) The impact of morphological richness and corpus size on embedding quality is empirically analyzed.
4) Practical insights for selecting appropriate embedding models for Uzbek NLP tasks are provided.

The remainder of the paper is organized as follows. Section 2 reviews related work on word embeddings and low-resource languages. Section 3 describes the proposed methodology and system architecture. Section 4 presents experimental results and analysis. Finally, Section 5 concludes the paper and outlines directions for future research.

2 RELATED WORK

Word embedding techniques have become a fundamental component of modern natural language processing systems. Early distributional models relied on co-occurrence statistics, while neural embedding approaches significantly improved semantic representation quality. Among these, Word2Vec introduced continuous vector representations by learning word semantics through local context prediction, demonstrating strong performance in various downstream tasks [1].

However, Word2Vec and similar static embedding models are known to suffer from limitations when applied to morphologically rich and low-resource languages. In such languages, a large number of word forms emerge from a single root due to extensive inflection and derivation, leading to severe data sparsity and out-of-vocabulary (OOV) issues. Several studies have highlighted that word-level embeddings trained on limited corpora struggle to generalize across unseen word forms [3], [6].

To address these challenges, subword-based embedding models were proposed. FastText extends Word2Vec by representing words as a collection of character n-grams, enabling the generation of embeddings for unseen or rare words. This approach has been shown to be particularly effective for agglutinative languages, where morphological variations are frequent [4]. Prior research demonstrates that FastText consistently improves OOV coverage and semantic robustness compared to traditional word-level embeddings [4].

More recently, contextual embedding models based on Transformer architectures have reshaped the NLP landscape [5], [7]. Models such as BERT produce dynamic representations that depend on the surrounding context, allowing the same word to have different embeddings in different sentences [5]. Multilingual variants, including mBERT, have enabled cross-lingual transfer and provided baseline performance for low-resource languages without requiring language-specific pretraining [8], [9]. Several studies report that contextual embeddings outperform static and subword-based models in sentence-level semantic tasks [7], [10].

Research on Uzbek language processing remains limited in comparison to high-resource languages, although recent studies on other low-resource language families have demonstrated the growing feasibility of neural NLP in underrepresented settings. Existing works primarily focus on rule-based methods, morphological analysis, or small-scale embedding experiments. Although recent efforts have explored neural approaches for Uzbek NLP, comprehensive comparative studies involving static, subword-aware, and contextual embeddings within a unified framework are still scarce.

In contrast to prior work, this study provides a systematic comparison of Word2Vec, FastText, and mBERT for the Uzbek language using a consistent experimental pipeline. By analyzing intrinsic evaluation metrics and examining the effects of morphological richness and corpus size, the proposed work contributes new empirical insights into embedding selection for low-resource and agglutinative languages.

3 METHODOLOGY AND SYSTEM ARCHITECTURE

This section describes the proposed experimental methodology and the overall system architecture used to train and evaluate embedding models for the Uzbek language. The methodology is designed to ensure reproducibility, fairness of comparison, and suitability for low-resource and morphologically rich language settings.

3.1 Data Collection and Corpus Preparation

The experiments are conducted on an Uzbek text corpus composed of educational, scientific, and technology-oriented texts. Due to the limited availability of large-scale publicly annotated datasets for the Uzbek language, the corpus is expanded using synthetically generated text used solely for data augmentation purposes, without including evaluation samples. This approach may help reduce data sparsity while preserving linguistic diversity [9].

All collected texts are converted into plain UTF-8 encoded format, where each line corresponds to a single sentence. The corpus is divided into raw and processed versions to clearly separate data acquisition from preprocessing steps.

Approximately 25% of the corpus consists of synthetically generated sentences used for data augmentation.

Synthetic data was generated using rule-based sentence templates combined with Uzbek morphological variations.

To avoid evaluation bias, none of the synthetic sentences were included in the evaluation dataset.

The synthetic portion was used only to enrich lexical and morphological diversity during training and was never used for intrinsic evaluation. The evaluation subset was constructed exclusively from non-synthetic held-out material in order to avoid leakage and overly optimistic similarity estimates.

3.2 Text Preprocessing

Text preprocessing plays a crucial role in ensuring the quality of learned embeddings. The preprocessing pipeline includes the following steps:

- Conversion of all text to lowercase.
- Removal of URLs and non-linguistic symbols.
- Preservation of Uzbek Latin and Cyrillic characters.
- Normalization of whitespace.
- Removal of extremely short or empty sentences.

The preprocessing procedure results in a clean and normalized corpus suitable for training embedding models. This step reduces noise and improves the consistency of semantic representations learned during training.

3.3 Static Embedding Models

3.3.1 Word2Vec

Word2Vec is employed as a baseline static embedding model in this study. The model is trained using the Skip-gram architecture, which aims to predict surrounding context words given a target word [1]. Formally, the Skip-Gram model maximizes the average log-probability of observing context words within a window of size m around a target word w_t:

$$L = \frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\le j\le m \\ j\neq 0}} log\ P(w_{t+j}|w_t), \qquad (1)$$

where T denotes the total number of words in the corpus, and $P(w_{t+j}|w_t)$ represents the conditional probability of a context word given the target word. This probability is typically modeled using the softmax function:

$$P(w_o|w_i) = \frac{\exp(V_{w_o}{}^{*}\, V_{w_i})}{\sum_{w=1}^{V}\exp(V_{w}{}^{*}\, V_{w_i})}, \qquad (2)$$

where V_{w_i} and V_{w_o} are the input and output vector representations of the target and context words, respectively, and V is the vocabulary size.

Skip-Gram is selected due to its effectiveness on relatively small corpora and its ability to capture fine-grained semantic relationships between words [1]. The training process adjusts word vectors such that semantically related words obtain similar representations in the embedding space.

The output of the Word2Vec model consists of fixed-dimensional word vectors, where each word is represented by a single embedding regardless of context. While computationally efficient, this approach is sensitive to vocabulary size and out-of-vocabulary (OOV) issues, particularly in morphologically rich languages such as Uzbek, where extensive inflection leads to a large number of word forms [1], [11].

3.3.2 FastText

FastText extends the Word2Vec framework by incorporating subword information through character n-grams, enabling richer morphological representation of words [4]. Instead of learning a single vector per word, FastText represents each word as a bag of its constituent character n-grams and computes the word embedding as the sum of the

corresponding n-gram vectors.

Formally, let $G(w)$ denote the set of character n-grams extracted from a word w. The embedding of a word w is defined as:

$$z_w = \sum_{g \in G(w)} z_g \quad (3)$$

where z_g is the vector representation of the n-gram g. Training is performed using the same Skip-Gram objective as in Word2Vec but applied to subword-based representations. The training objective therefore becomes:

$$L = \frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m \le j \le m \\ j \neq 0}} log\ P(w_{t+j}|w_t) \quad (4)$$

with the conditional probability modeled as:

$$P(w_o|w_i) = \frac{\exp(z_{w_o} * z_{w_i})}{\sum_{w=1}^{V}\exp(z_w * z_{w_i})} \quad (5)$$

where z_{w_i} and z_{w_o} are subword-composed embeddings of the target and context words, respectively.

This property makes FastText particularly suitable for morphologically rich and agglutinative languages such as Uzbek, where extensive affixation leads to a large number of word variants [4], [6]. By leveraging subword information, FastText significantly improves vocabulary coverage, enhances robustness to out-of-vocabulary (OOV) words, and reduces the negative impact of data sparsity compared to purely word-level embeddings [4].

3.4 Contextual Embedding Model

3.4.1 Multilingual BERT (mBERT)

To capture contextual semantics beyond word-level representations, multilingual BERT (mBERT) is employed as a contextual embedding model [8]. Unlike static embeddings, mBERT generates dynamic representations that depend on the surrounding context of each sentence.

In this study, mBERT is used to produce sentence-level embeddings by applying a mean pooling strategy over token representations from the final hidden layer. The resulting embeddings have a dimensionality of 768 and encode rich semantic and syntactic information. Fine-tuning is not performed in order to maintain model stability and reduce computational complexity, which aligns with realistic low-resource research constraints (Table 1).

Table 1: Training hyperparameters of embedding models.

Model	Dimen-sion	Window	Min Count	Epochs
Word2-Vec	300	5	5	10
Fast-Text	300	5	5	10
mBERT	768	-	-	Pretrained

Note: mBERT is a pretrained model and does not require training hyperparameters.

3.5 Evaluation Methodology

The evaluation protocol was designed to ensure reproducibility and to distinguish between word-level and sentence-level semantic assessment. For Word2Vec and FastText, intrinsic evaluation was conducted at the word level using three indicators: vocabulary size, OOV coverage, and average cosine similarity. OOV coverage was computed on a held-out list of Uzbek word forms that were excluded from the training vocabulary and selected to reflect morphological variation, including inflected, derived, and less frequent forms. A token was considered covered if the corresponding model was able to generate a valid embedding representation for that item.

Average cosine similarity for Word2Vec and FastText was calculated over a manually curated set of semantically related Uzbek word pairs. The purpose of this metric was not to establish an absolute benchmark score, but to provide a controlled relative comparison between static and subword-aware embeddings on the same lexical material. Because the evaluation set was limited and intentionally composed of related pairs, high cosine values were expected; therefore, the scores should be interpreted as relative similarity indicators rather than as evidence of universally strong semantic performance.

For mBERT, evaluation was performed separately at the sentence level, since contextual transformer embeddings are not directly comparable to static word embeddings on the same representation basis (Table 2). Sentence embeddings were obtained by mean pooling over the final hidden-layer token representations. Cosine similarity was then computed over selected Uzbek sentence pairs reflecting semantic relatedness at the contextual level. Consequently, the results of Word2Vec/FastText and mBERT are reported in separate tables and interpreted within their respective representation levels.

Table 2: Summary of the evaluation protocol.

Component	Word2Vec	FastText	mBERT
Representation level	Word	Word/subword	Sentence
Evaluation unit	Word pairs	Word pairs	Sentence pairs
OOV test set	Yes	Yes	No
Cosine similarity basis	Related word pairs	Related word pairs	Related sentence pairs
Directly comparable with others	Word-level only	Word-level only	Sentence-level only

3.6 System Architecture Overview

The overall system architecture follows a modular pipeline consisting of data preprocessing, model training, embedding generation, and evaluation. Each component is implemented as an independent module, enabling flexible experimentation and reproducibility. The complete pipeline is publicly available through an open-source repository, ensuring transparency and facilitating future extensions of the proposed framework.

4 EXPERIMENTAL RESULTS AND DISCUSSION

This section presents the experimental results obtained from the comparative evaluation of Word2Vec, FastText, and mBERT embedding models for the Uzbek language. The analysis focuses on intrinsic evaluation metrics to assess the effectiveness of each approach in handling low-resource and morphologically rich language characteristics.

4.1 Quantitative Results

The performance of the embedding models is evaluated using vocabulary size, out-of-vocabulary (OOV) coverage, and average cosine similarity. Table 3 summarizes the experimental results obtained from the initial corpus.

The intrinsic evaluation results of the embedding models are presented in Table 3.

Because antonym pairs, unrelated pairs, and harder semantic distinctions were not included in the present intrinsic set, the cosine values should not be interpreted as broad semantic benchmark scores; rather, they reflect relative behavior of the evaluated models on a restricted related-pair sample.

Word2Vec shows limited ability to represent unseen tokens, resulting in lower OOV coverage compared to FastText.

The high cosine similarity values (close to 1.0) are primarily due to the limited size of the evaluation set and the use of semantically related word pairs. These values therefore indicate relative similarity trends rather than absolute semantic performance.

Table 3: Intrinsic evaluation results of embedding models on the Uzbek corpus.

Model	Vocabulary Size	OOV Coverage	Avg Cosine Similarity
Word2Vec	1760	0.310	0.997
FastText	1760	≈1.000	0.989

Note: OOV coverage for FastText approaches 1.0 due to subword modeling.

4.2 Analysis of Static and Subword Embeddings

The observed superiority of FastText in OOV handling highlights the importance of subword modeling for the Uzbek language. Due to its agglutinative morphology, Uzbek words often appear in numerous inflected forms, many of which may not be explicitly present in the training corpus. Word2Vec, which relies solely on word-level representations, is unable to generalize across these variations.

4.3 Contextual Embedding Results

Due to its contextual and sentence-level nature, mBERT is evaluated separately from word-level embedding models. Sentence embeddings were generated using the multilingual BERT model (bert-base-multilingual-cased) with mean pooling over the final hidden layer representations.

In addition to static and subword-based embeddings, mBERT is employed to generate contextual sentence-level representations. Unlike Word2Vec and FastText, mBERT produces dynamic embeddings that capture semantic meaning conditioned on context. The generated sentence embeddings have a dimensionality of 768, enabling richer semantic encoding.

Qualitative inspection of cosine similarity scores between selected sentence embeddings reveals that mBERT effectively captures contextual relationships that are not accessible to static models. This advantage is particularly relevant for sentence-level tasks such as semantic similarity, classification, and information retrieval.

In addition to qualitative analysis, a quantitative sentence-level evaluation was performed for the multilingual BERT (mBERT) model. Sentence embeddings were generated using the bert-base-multilingual-cased model, and cosine similarity was calculated between sentence pairs to estimate semantic similarity.

Table 4 presents the sentence-level evaluation results for the multilingual BERT (mBERT) model, measured using average cosine similarity between sentence embeddings.

Table 4: Sentence-level similarity evaluation using multilingual BERT.

Model	Embedding Dimension	Avg Cosine Similarity
mBERT	768	0.822

The results indicate that multilingual BERT is capable of capturing contextual semantic relationships at the sentence level. Compared to static embedding models such as Word2Vec and FastText, mBERT provides richer contextual representations because the embedding of each token depends on the surrounding sentence context.

4.4 Comparative Discussion

The experimental findings suggest that each embedding approach offers distinct advantages depending on the linguistic and computational constraints. Word2Vec serves as a lightweight baseline but struggles with morphological variability and unseen words. FastText provides a robust improvement by incorporating subword information, making it a more suitable choice for Uzbek word-level representations. mBERT, while computationally more demanding, delivers the most expressive representations by modeling contextual semantics at the sentence level.

These results confirm that no single embedding model is universally optimal. Instead, the choice of embedding technique should be guided by the target task, available computational resources, and the linguistic properties of the language. For low-resource and morphologically rich languages such as Uzbek, subword-aware and contextual embeddings offer clear advantages over purely static approaches.

Direct comparison between static word embeddings (Word2Vec and FastText) and contextual embeddings (mBERT) is not strictly equivalent because the former operate at the word level while the latter operates at the sentence level.

Therefore, results are interpreted within their respective representation levels rather than as a direct metric comparison.

The comparison between Word2Vec/FastText and mBERT is intentionally reported in a non-identical evaluation setting because these models operate on different representation levels. Word2Vec and FastText are designed primarily for lexical representations, whereas mBERT captures sentence-dependent contextual semantics. Therefore, the present study does not claim direct metric equivalence across all three models; instead, it provides a structured comparison of their practical suitability for Uzbek NLP tasks at different representation levels.

5 CONCLUSIONS

This paper presented a comparative study of static, subword-aware, and contextual embedding models for the Uzbek language within a unified and reproducible experimental framework. The study focused on evaluating Word2Vec, FastText, and multilingual BERT (mBERT) with respect to their ability to represent semantic information in a low-resource and morphologically rich language setting.

The experimental results demonstrate that traditional static embeddings such as Word2Vec are limited in handling out-of-vocabulary words and morphological variability, which are common characteristics of the Uzbek language. In contrast, FastText significantly improves vocabulary coverage by incorporating subword information, making it a more suitable choice for word-level representation tasks [6]. Furthermore, mBERT provides the most expressive representations by capturing contextual semantics at the sentence level, highlighting the advantages of transformer-based models for deeper language understanding.

The findings of this study emphasize that embedding selection should be guided by both linguistic properties and task requirements. While FastText offers a balanced trade-off between performance and computational efficiency, contextual models such as mBERT are particularly beneficial for applications that require sentence-level or context-dependent understanding.

6 FUTURE WORK

Future work will focus on expanding the size and diversity of the Uzbek text corpus to further enhance embedding quality. Additionally, fine-tuning mBERT on domain-specific Uzbek data and evaluating the models on downstream tasks such as text classification [7], information retrieval, and question answering constitute promising directions for further research. Multilingual variants, including mBERT, have enabled cross-lingual transfer and demonstrated the value of multilingual pretraining for low-resource languages [4], [13].

REFERENCES

[1] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," arXiv preprint arXiv:1301.3781, 2013, [Online]. Available: https://doi.org/10.48550/arXiv.1301.3781.

[2] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation," in Proc. EMNLP, 2014, pp. 1532-1543, [Online]. Available: https://doi.org/10.3115/v1/D14-1162.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in Proc. NAACL-HLT, 2019, pp. 4171-4186, [Online]. Available: https://doi.org/10.18653/v1/N19-1423.

[4] T. Pires, E. Schlinger, and D. Garrette, "How Multilingual Is Multilingual BERT?" in Proc. ACL, 2019, pp. 4996-5001, [Online]. Available: https://doi.org/10.18653/v1/P19-1493.

[5] M. E. Peters et al., "Deep Contextualized Word Representations," in Proc. NAACL-HLT, 2018, pp. 2227-2237, [Online]. Available: https://doi.org/10.18653/v1/N18-1202.

[6] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," Transactions of the Association for Computational Linguistics, vol. 5, pp. 135-146, 2017, [Online]. Available: https://doi.org/10.1162/tacl_a_00051.

[7] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of Tricks for Efficient Text Classification," in Proc. EACL, 2017, pp. 427-431, [Online]. Available: https://doi.org/10.18653/v1/E17-2068.

[8] A. Rogers, O. Kovaleva, and A. Rumshisky, "A Primer in BERTology: What We Know About How BERT Works," Transactions of the Association for Computational Linguistics, vol. 8, pp. 842-866, 2020, [Online]. Available: https://doi.org/10.1162/tacl_a_00349.

[9] R. Cotterell et al., "On the Complexity and Typology of Inflectional Morphological Systems," Transactions of the Association for Computational Linguistics, vol. 6, pp. 327-342, 2018, [Online]. Available: https://doi.org/10.1162/tacl_a_00026.

[10] Y. Goldberg, Neural Network Methods for Natural Language Processing. San Rafael, CA, USA: Morgan & Claypool, 2017, [Online]. Available: https://doi.org/10.2200/S00762ED1V01Y201703HLT037.

[11] D. Jurafsky and J. H. Martin, Speech and Language Processing, 3rd ed., draft, 2023, [Online]. Available: https://doi.org/10.48550/arXiv.2308.08246.

[12] M. Artetxe, S. Ruder, and D. Yogatama, "On the Cross-lingual Transferability of Monolingual Representations," in Proc. ACL, 2020, pp. 4623-4637, [Online]. Available: https://doi.org/10.18653/v1/2020.acl-main.421.

[13] G. Lample and A. Conneau, "Cross-lingual Language Model Pretraining," in Advances in Neural Information Processing Systems (NeurIPS), vol. 32, 2019, [Online]. Available: https://doi.org/10.48550/arXiv.1901.07291.

APPENDIX

A) Experimental Implementation Details (Code-Centric)

This appendix emphasizes the software implementation, code structure, and reproducibility aspects of the conducted experiments. The study was designed as a fully code-driven and reproducible workflow, where all experimental results reported in the paper were generated automatically from scripts without manual intervention.

B) Project Structure and Code Organization

The experimental framework was implemented as a modular Python project with a clear separation of responsibilities. The repository structure follows a component-oriented design, where data acquisition, preprocessing, machine learning models, optimization modules, and visualization components are organized into independent but interconnected blocks. The overall software organization and module relationships are illustrated in Figure A1.

```
uzbek-hybrid-embeddings/
├── data/
│   ├── raw/              # Original Uzbek text files
│   └── processed/        # Cleaned and tokenized corpus
├── src/
│   ├── preprocessing.py    # Text cleaning and normalization
│   ├── train_word2vec.py   # Word2Vec training script
│   ├── train_fasttext.py   # FastText training script
│   ├── evaluate.py         # Evaluation and comparison metrics
│   └── visualize.py        # Tables and plots generation
├── results/
│   ├── tables/           # CSV evaluation results
│   └── figures/          # Generated plots
├── README.md             # Reproducibility instructions
└── requirements.txt      # Dependency list
```

Figure A1: Modular software architecture of the CACIP experimental framework.

This structure ensures extensibility, allowing new embedding models or evaluation metrics to be added with minimal changes to the existing codebase.

C) Text Preprocessing Implementation

All textual data preprocessing was performed programmatically using a dedicated script (preprocessing.py). The preprocessing pipeline includes lowercasing, removal of URLs and special characters, preservation of Uzbek-specific letters, and whitespace normalization.

Each preprocessing step is explicitly implemented in code, ensuring that the same corpus can be regenerated deterministically from raw data. This design eliminates hidden preprocessing bias and supports reproducible experimentation.

D) Embedding Training Scripts

Each embedding model is trained using an independent Python script to ensure isolation and transparency:

- train_word2vec.py implements a skip-gram Word2Vec model using the Gensim library.
- train_fasttext.py trains a FastText model with subword n-grams enabled to handle morphological variation.
- mBERT embeddings are generated through the HuggingFace Transformers interface without fine-tuning, using sentence-level pooling strategies implemented directly in code.

Model hyperparameters (embedding dimension, window size, minimum frequency, epochs) are explicitly defined in the scripts to facilitate reproducibility and fair comparison.

Due to file size constraints, trained models are excluded from version control and are regenerated locally using the provided scripts.

E) Evaluation and Result Generation Code

The evaluation process is entirely script-based and implemented in evaluate.py. The script computes intrinsic evaluation metrics, including:

- Vocabulary size.
- Out-of-vocabulary (OOV) coverage.
- Average cosine similarity.

All reported tables are automatically generated as CSV files, while visualizations are produced programmatically using Matplotlib. This approach ensures that any change in data or model configuration is consistently reflected in the results.

F) Reproducibility and Open-source Availability

To promote transparency and reproducible research, the complete experimental pipeline - including preprocessing, training, evaluation, and visualization code - has been made publicly available. GitHub repository: The full experimental pipeline and implementation code are publicly available at:
https://github.com/Abdilatif1909/uzbek-hybrid-embeddings