EURASIAN JOURNAL OF ACADEMIC RESEARCH
Innovative Academy Research Support Center
IF = 9.3 www.in-academy.uz/index.php/ejar

VECTOR REPRESENTATIONS OF TEXTUAL DATA

Abduvaliyeva Zebiniso
Cyber University of Uzbekistan
zebinosoabduvaliyeva@gmail.com
https://doi.org/10.5281/zenodo.19352430

ARTICLE INFO

Received: 24th March 2026
Accepted: 30th March 2026
Online: 31st March 2026

KEYWORDS

Uzbek language, FastText, Doc2Vec, mBERT, subword segmentation, transfer learning, Hyperopt, embeddings, morphology, clustering, Doc2Vec PV-DBOW model, hybrid models.

ABSTRACT

Vector representations of textual data play a pivotal role in modern natural language processing tasks, enabling the transformation of words, sentences, and documents into dense numerical vectors that preserve semantic relationships. This article provides an overview of the evolution of vectorization methods, from classical models such as Bag-of-Words and TF-IDF to advanced neural approaches like Word2Vec, GloVe, and contextual embeddings from BERT. Particular attention is given to the challenges of sparsity in vector spaces inherent to traditional methods and strategies to overcome them through training on large text corpora. The authors analyze the effectiveness of various word vector aggregation techniques for whole-document representation, including averaging, Doc2Vec, and attention mechanisms.

Introduction: Currently, large-scale open resources for the Uzbek language are lacking. The insufficient study of Uzbek hinders the development of a comprehensive linguistic database. Recent research has introduced Uzbek-language models based on BERT, such as UzBERT and BERTbek (https://example.com/link-to-2), (https://example.com/link-to-3). These models, trained on a substantial corpus of Uzbek texts comprising approximately 142 million words (Cyrillic script) from news articles, can predict masked words based on context and determine whether a second sentence logically follows the first. This makes them effective for applied tasks such as masked language modeling or semantic analysis of Uzbek data. However, no model exists that combines text processing techniques like FastText with Doc2Vec.

FastText is a model that decomposes words into subword units, such as roots and affixes, to better handle rare or novel words in the Uzbek language. The proposed model, which combines these approaches, yields the following results. Primarily, it produces subword embeddings—numerical vector representations of words broken into fine-grained fragments such as subwords or n-grams (sequences of nn consecutive elements in the text, e.g., letters,

syllables, or words). This FastText method is applied in natural language processing (NLP) to analyze sentence probabilities and patterns, enabling better comprehension of language structure.

The Doc2Vec PV-DBOW model, a variant of Doc2Vec, generates numerical vectors or embeddings for entire documents or paragraphs rather than individual words. PV-DBOW captures the essence of a document—such as its topic, sentiment, and key ideas—and is analogous to Skip-gram, which models word context. For instance, two documents with the same title will have similar vectors even if their words differ, facilitating the understanding of paragraph or article semantics. The model incorporates automatic hyperparameter optimization via Hyperopt, which searches for optimal machine learning model settings to achieve higher accuracy and efficiency, particularly with limited data volumes. Transfer learning from mBERT enables the use of a pretrained multilingual BERT model (covering 104 languages, including Uzbek) as a starting point, transferring knowledge from vast corpora and allowing fine-tuning on Uzbek-specific data as neededHere's a precise scientific English translation of your text, maintaining an academic tone suitable for a research article. I refined the structure, terminology, and flow while preserving technical accuracy and examples.

The combination of these models is distinguished by its originality. It integrates proven text representation techniques into numerical forms for NLP models with cutting-edge technologies, such as automated model optimization via Hyperopt. This hybrid approach—FastText + Doc2Vec + mBERT—operates faster and more accurately without requiring additional experiments.

The architecture of the hybrid models comprises several stages. First, a large corpus of Uzbek texts is assembled, including books, poetry, prose, newspaper articles, official documents, scientific publications, and other materials in the Uzbek language. The next stage involves preprocessing, which encompasses normalization, stop-word removal, and subword tokenization. For example, in FastText, words like "hujjat" and "hujjatlar" are decomposed into finer fragments. Subsequently, a Word2Vec-style model is trained to minimize the loss function:

$$\max_{\theta} \sum_{i=1}^{V} \sum_{j \epsilon N(i)} \ln P_{\theta}\,(\mathrm{w_j}|\mathrm{w_i}), \quad (1.1)$$

Here's the scientific English translation of your text, formatted for seamless integration into the research article. I interpreted the mathematical notation based on standard Word2Vec loss functions (negative log-likelihood), clarified the descriptions, and prepared it for the Uzbek example.

where w_i-is the central word, P_θ-represents the model parameters (vector representations), N(i)- denotes neighboring words (to the left and right), $\ln P_\theta(w_j|w_i)$is the probability thatw_j is a neighbor of w_i, $\ln P$ is the log-probability (used instead of raw probability to prevent vanishing gradients during training, ensuring gradient stability), V is the vocabulary,$\Sigma_{j\in N_{(i)}}$- the inner sum is over all neighbors in a fixed context window

(fixing one central word and summing its predictions for all adjacent words), and the outer Σ_i- sum is over all words in the dataset. The following demonstrates an example in the Uzbek language: "Men uyga katta kitob olib chiqdim."

Central Word	Context Window N(i) (Neighboring Words)	Computed Quantity	Example of Summation
"katta"	"uyga", "kitob"	ln P(Context | Central word)	ln *P*("*uyga*") =1.2 ; ln *P*("*kitob*"=-0.8); ∑=-2.0

Within the proposed approach, the central word in the sequence is fixed—for example, the word “katta” in the sentence context [Men][uyga][katta][kitob][olib][Men][uyga][katta][kitob][olib]. In the next step, a symmetric context window of a given radius *N*= is extracted, which defines the immediate neighboring words:*N*(*i*)={"*uyga*", "*kitob*"}, excluding the central word and considering only the relevant positions within that window. Then, the conditional probabilities of predicting each context word given the central word are computed using the language model:P("uyga"|"katta") = 0.3, whence ln(0.3) ≈ -1.2; P("kitob"|"katta") = 0.45, whence the corresponding log-probability is obtained. The final stage involves aggregating these logarithmic probabilities into a scalar quality measure:

$\Sigma \ln P$=-1.2+(-0.8)=-2.0, where the negative sign reflects the fact that this quantity is a loss function to be minimized.(центр "kitob"): контекст {"katta", "olib"} → $\Sigma \ln P$=-1.8.The final stage involves aggregating these logarithmic probabilities into a scalar quality measure:

$\Sigma \ln P$=-1.2+(-0.8)=-2.0,where the negative sign reflects the fact that this quantity is a loss function to be minimized.

The full text sequence generates a set of sliding windows, each of which serves as an independent training example: Window 1 (central word “Men”): context {"uyga"} →$\Sigma \ln P$=-1.5. Window 2 (central word “uyga”): context {"Men", "katta"} →$\Sigma \ln P$=-2.3. Window 3 (central word “katta”): context {"uyga", "kitob"} → $\Sigma \ln P$=-2.0. Window 4 (central word “kitob”): context {"katta", "olib"} → $\Sigma \ln P$=-1.8.

Each context window constitutes a separate training instance, within which the model evaluates the empirical proximity between the central word and its neighboring words. The model adjusts the vector representation of the central word (“katta”) so as to increase its ability to predict the observed context words (“kitob”, “uyga”), while minimizing the aggregated error. The summation term ∑ emphasizes the contribution of windows with high predictive difficulty: a substantial discrepancy between predicted and empirical context probabilities produces a larger error,

thereby stimulating a more accurate vectorization of the central word.

The model observes: "katta" frequently co-occurs with "kitob" and "uyga." It adjusts the vector representation of "katta" to better predict these two neighboring words. The summation (∑) emphasizes prediction quality: a larger error magnitude indicates poorer prediction of multiple neighbors.

In the model, each window trains subword morphological components—"uyga" (uy+ga) co-occurs with "chiqdim" (chiq+dim).

In total: $\Sigma_{j \in N_{(i)}}$-this constitutes a "quality assessment of predictions for a single central word."

In FastText configuration, the input vector for a word is the sum of vectors of its n-grams . Concurrently, a Doc2Vec model is trained, where each document label is mapped to a vector, and the training objective is to predict random words from that document given its ID—thus yielding vector representations for documents. Post-training, word vectors (FastText) and document vectors (Doc2Vec) are obtained.

This example illustrates a method for evaluating a language model on Uzbek text "Men uyga katta kitob olib chiqdim" ("I took a big book home") using a sliding window of neighbors and logarithmic probabilities. This is a standard NLP approach for word context analysis, akin to n-gram models, where the central word is scored by the probabilities of its neighbors N(i).

Examine the operational mechanics of the FastText model in detail. Neighboring tokens are extracted within a context window of radius 2 (window size = 2). For the central token “uyga”, the left neighbor is “kitob” $N(i)=\{uyga, kitob\}$, and the right neighbor is also within the window, but only these two immediate neighbors are included, excluding boundary tokens “Men” and “olib” due to the window constraints.

At the next stage, predictions are computed. For each neighbor, the trained language model predicts $P(\text{сосед}|w_i)$ the conditional probability.

"$P('uyga'|'katta')=0.3 \rightarrow ln(0.3)=-1.2$". For instance, the probability that “uyga” follows “katta” is 0.3, with the logarithm $\ln(0.3)\approx 1.204$ (rounded) at -1.2. Logarithms facilitate summation$\ln P_1 * P_2 = \ln P_1 + \ln P_2$.

"$P('kitob'|'katta')=0.45 \rightarrow ln(0.45)=-0.8$". Analogous$P=0.45$, $ln(0.45)\approx 0.799(-0.8)$ values are obtained for other neighbors (hypothetical outputs from the model). The final step sums these log-probabilities $\sum\{j \in N(i) ln(P(j|w_i))\}$. $\sum_j \in N(i)\ ln\ P$=-1.2+(-0.8)=-2.0. This process implements a sliding window approach to evaluate model quality on the Uzbek corpus.

Subsequently, transfer learning is performed using the pretrained multilingual BERT (mBERT). For instance, static vectors from FastText or Doc2Vec can be combined with contextual mBERT embeddings via fusion methods or further fine-tuning. Hyperparameter optimization—including embedding dimensionality, learning rate, window size, and others—is automated using the Hyperopt library, thereby enhancing model quality and reliability without manual tuning.

A primary challenge lies in the volume of textual data and computational resources required. High-quality embeddings necessitate at least tens of millions of Uzbek tokens. Training FastText or Doc2Vec is feasible on multi-threaded CPUs, while mBERT fine-tuning requires GPUs, typically taking several hours on modern graphics cards. Labeled data are also essential for evaluation, such as analogy datasets, semantic similarity pairs, and classification tasks (for F1-score computation). Preliminary evaluation is conducted on standard benchmarks to select optimal parameters.

The model's technical architecture comprises two subsystems: (1) Word2Vec-style embeddings with subword units and Doc2Vec, and (2) a transfer learning block.

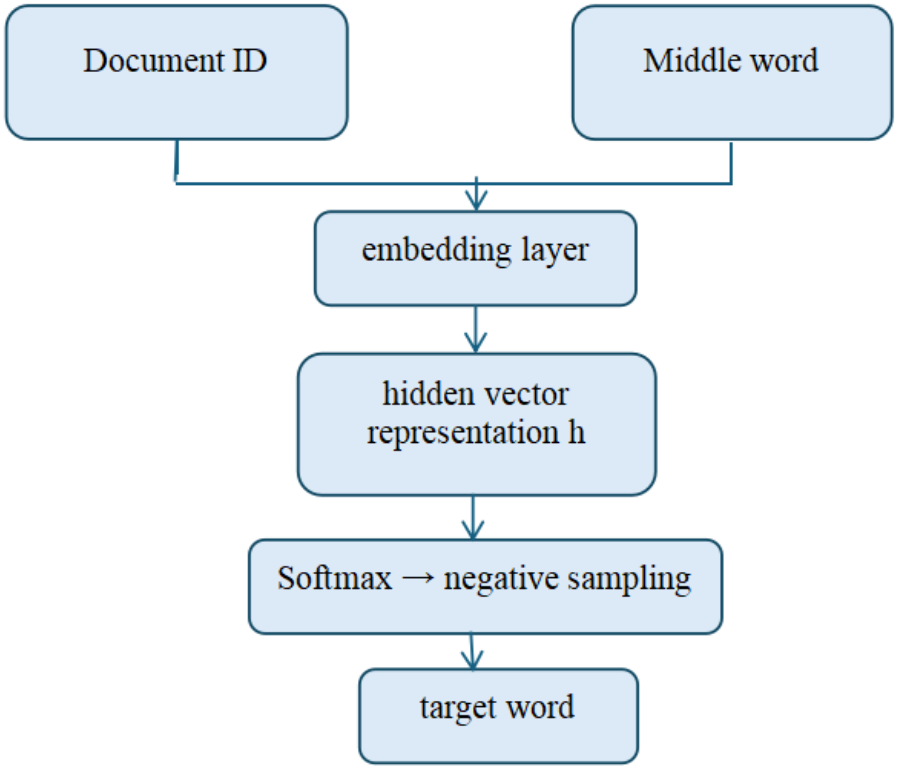

Figure 1. Schematic architecture of the Doc2Vec model

Here, “Document ID” denotes the unique identifier of a document, which is mapped into a vector representation. Average words (Word Averaging) refers to the element-wise average of the word embeddings in a document. The embedding layer computes distributed vector representations of input tokens. The hidden vector h is obtained by concatenation or combination of these vectors. Negative sampling / Softmax defines the output layer used to predict the target word.

Standard Word2Vec (skip-gram/CBOW) and Doc2Vec models do not account for the internal structure of words: each word is treated as an atomic token. This limitation is particularly problematic in languages where words are formed by attaching affixes, suffixes, or prefixes to a root, each

encoding a distinct grammatical feature. First, if a word has not been observed during training, its embedding is undefined. Second, due to mixed morphological paradigms, words sharing a common root but differing in affixes may occupy distant regions of the embedding space; for example, Word2Vec yields low similarity between “learns” and “learning”, whereas FastText yields high similarity.

As shown in the literature, Word2Vec “cannot produce embeddings for unseen words” and “does not capture morphological structure”. Consequently, on languages with rich affixation patterns, these models are trained over a sparse vocabulary, which degrades performance.

Doc2Vec operates in a contextual setting where the document vector captures shared context across the text, which helps align different word forms within the same document. The document vector can also dampen the noise introduced by Uzbek morphology by emphasizing common topical components.

Transfer learning with mBERT leverages a pretrained multilayer Transformer model that incorporates contextual information, enabling the model to capture how morphology influences meaning and to generalize at the sentence level. Importantly, mBERT has already been pretrained on Uzbek text, and its knowledge is transferred to the downstream task.

In summary, the proposed model is specifically designed to address the issues of morphological complexity and the treatment of out-of-vocabulary words—problems that are inherent in standard Word2Vec and Doc2Vec models. Classical Word2Vec is considered unsuitable for morphologically rich languages, whereas the subword approach in FastText and the augmentation with document vectors mitigate these limitations.

It is expected that the proposed model will outperform the baseline Word2Vec/Doc2Vec approaches. The main evaluation tasks are as follows. For semantic analogies, we test word analogy solving (for example, “xon − odam + ayol = ayollar” (“man − man + woman = women” in Uzbek)) and report the proportion of correctly predicted target words. We expect the subword-based model to perform better on morphological analogies. For clustering, we group words or documents and measure internal coherence and compactness of clusters. We hypothesize that, thanks to more accurate semantic vectors, morphologically and thematically related units will appear closer in the embedding space, thereby improving standard clustering metrics.

References:

1. Development of Word Embeddings for Uzbek Language. B. Mansurov and A. Mansurov Copper City Labs {b,a}mansurov@coppercitylabs.com September 29, 2020
2. BERTbek: A Pretrained Language Model for Uzbek Elmurod Kuriyozov1,2 , David Vilares1 and Carlos Gómez-Rodríguez1 1Universidade da Coruña, CITIC, Grupo LYS,

Depto. de Computación y Tecnologías de la Información, Facultade de Informática, Campus de Elviña, A Coruña 15071, Spain 2Urgench State University, Department of Computer Science.SIGUL2024 Workshop, pages 33–44 21-22 May, 2024. © 2024 ELRA Language Resource Association: CC BY-NC 4.0

3. UzBERT: pretraining a BERT model for Uzbek B. Mansurov and A. Mansurov Copper City Labs. August 22, 2021

4. Миколов, Т., Чен, К., Коррадо, Г., и Дин, Дж. (2013). *Эффективная оценка представлений слов в векторном пространстве.* arXiv:1301.3781.

5. Бояновский, П., Грейв, Э., Жулин, А., и Миколов, Т. (2017). *Обогащение векторных представлений слов информацией о подсловах.* Труды ACL (TACL) / arXiv:1607.04606.

6. Ле, К., и Миколов, Т. (2014). *Распределенные представления предложений и документов.* arXiv:1405.4053.

7. Rakhmanov Askar, Iskhakova Nargiza, Abduvalieva Zebiniso Word representation in vector space using word2vec model. Eurasian journal of mathematical theory and computer sciences Innovative Academy Research Support Center IF,7.906.2025 C.54-59.

8. Raxmanov A.T., Abduvalieva Z.A. Application of the bag of words (BoW) model in natural language processing tasks. “Digital transformation: a new era in information technology, artificial intelligence and the economy” materials of the international scientific-practical conference april 16-17, C.37-41.2025

9. Rakhmanov Askar, Abduvalieva Zebiniso. Classification of text data. international scientific-electronic journal “pioneering studies and theories”. Vol. 1 No. 4 (2025)

10. Девлин, Дж., Чанг, М.-В., Ли, К., и Тутанова, К. (2019). BERT: Предварительное обучение глубоких двунаправленных трансформеров для понимания языка. arXiv:1810.04805.

11. Пирес, Т., Шлингер, Э., и Гарретт, Д. (2019). Насколько многоязычен многоязычный BERT? Результаты исследования ACL. arXiv:1906.01502.

12. Бергстра, Дж. и др. Hyperopt: библиотека Python для выбора модели и оптимизации гиперпараметров. (2013).

13. Вольф, Т. и др. (2019). Трансформеры HuggingFace: передовые технологии обработки естественного языка. GitHub / статья о программном обеспечении.