*Acta Informatica Pragensia*
2026, Volume 15, Issue 2, 400–415
https://doi.org/10.18267/j.aip.309

PRAGUE UNIVERSITY OF ECONOMICS AND BUSINESS

**Article** Open Access

# Effect of Dimension Size and Window Size on Word Embedding in Classification Tasks

**Dávid Držík** [1], **Jozef Kapusta** [1,2]

[1] Faculty of Natural Sciences and Informatics, Constantine the Philosopher University in Nitra, Nitra, Slovakia
[2] Institute of Security and Computer Science, University of the National Education Commission, Krakow, Poland

Corresponding author: Jozef Kapusta (jkapusta@ukf.sk)

**Editorial Record**

**First submission received:**
July 17, 2025

**Revisions received:**
December 19, 2025
February 18, 2026

**Accepted for publication:**
February 23, 2026

**Academic Editor:**
Zdenek Smutny
Prague University of Economics and Business, Czech Republic

This article was accepted for publication by the Academic Editor upon evaluation of the reviewers' comments.

**How to cite this article:**
Držík, D., & Kapusta, J. (2026). Effect of Dimension Size and Window Size on Word Embedding in Classification Tasks. *Acta Informatica Pragensia*, 15(2), 400–415. https://doi.org/10.18267/j.aip.309



## Abstract

**Background:** Static word embedding models such as Word2Vec and GloVe remain widely used in natural language processing, yet key hyperparameters are often selected heuristically rather than through systematic validation.

**Objective:** This study provides an extrinsic evaluation of context window size and embedding dimensionality for Word2Vec (CBOW and Skip-gram) and GloVe embeddings in a downstream spam classification task.

**Methods:** Embeddings were trained on a large external corpus and evaluated using a neural network and several classical machine learning classifiers.

**Results:** The results show that context window size has a moderate influence on performance, whereas embedding dimensionality has a clearer effect: values below approximately 50 degrade performance, while increases beyond moderate ranges (approximately 100–150) yield diminishing returns. Across all experiments, Word2Vec achieves higher stability and performance than GloVe.

**Conclusion:** Overall, the findings suggest that robust classification performance can be achieved with moderate embedding dimensionalities and smaller context windows, providing practical guidance for efficient embedding configuration.

## Index Terms

Word embeddings; Word2Vec; GloVe; Vector dimension; Context window size.

## 1 INTRODUCTION

Artificial intelligence (AI) methods have become an integral component of a wide range of application domains, with natural language processing (NLP) playing a central role in tasks involving textual data. Since most machine learning algorithms operate on numerical inputs, textual information must first be transformed into a suitable numerical representation. In NLP, this transformation is commonly achieved through word embeddings, which represent words as continuous vectors in a high-dimensional space, capturing semantic and syntactic properties of language (Dharma et al., 2022; Khurana et al., 2023; Wyatt et al., 2021).

Among embedding techniques, predictive and count-based models such as Word2Vec (Mikolov, Chen, et al., 2013) and GloVe (Pennington et al., 2014) remain widely used due to their conceptual simplicity, computational efficiency, and applicability to downstream tasks. In the Word2Vec framework, word meaning is inferred from contextual co-occurrence patterns, resulting in vector representations that preserve linguistic regularities and semantic relationships.

Distances between vectors in the embedding space often correspond well to human judgments of word similarity (Abubakar et al., 2022; Mikolov, Chen, et al., 2013).

Although more recent contextualized language models such as BERT (Devlin et al., 2019), GPT (Brown et al., 2020), and ELMo have demonstrated superior performance across many NLP benchmarks, they differ fundamentally from Word2Vec and GloVe in terms of representation strategy. Contextualized models generate dynamic embeddings that depend on sentence-level context, requiring access to the full model during inference. In contrast, Word2Vec and GloVe produce static embeddings that can be precomputed and reused across tasks without retraining the embedding model. This characteristic makes static embeddings particularly attractive in scenarios where computational efficiency, reproducibility, or deployment constraints are critical.

Despite their widespread adoption, the performance of Word2Vec and GloVe embeddings is highly sensitive to hyperparameter choices, most notably the context window size and the dimensionality of the embedding space. Suboptimal parameter settings can lead to degraded embedding quality and reduced downstream task performance. Prior studies have explored the impact of these parameters, often reporting markedly different recommendations. For example, Adewumi et al. (2020) investigated dimensionalities ranging from 100 to 3000 and window sizes of 4 and 8, concluding that optimal configurations are task-dependent and that increasing dimensionality beyond 400 may even deteriorate performance. Similarly, Yang et al. (2018) reported optimal results for a Twitter classification task using an embedding dimensionality of 800 and a window size of 10. Nazir et al. (2022), focusing on Urdu word embeddings, identified a dimensionality of 500 and a window size of 5 as optimal based on intrinsic evaluation metrics.

While these studies provide valuable insights, their conclusions exhibit substantial variability and are often derived from task-specific or intrinsic evaluations. Consequently, there is no clear consensus regarding whether high-dimensional embeddings are necessary for practical downstream classification tasks. Moreover, the commonly recommended dimensionalities reported in the literature frequently exceed what is typically used in applied NLP pipelines, raising questions about their computational efficiency and general applicability.

Motivated by these observations, this study aims to provide a systematic empirical validation of Word2Vec and GloVe hyperparameters under an extrinsic evaluation setting. Rather than proposing a novel embedding model, we focus on assessing the practical necessity of commonly recommended parameter ranges by evaluating their impact on downstream classification performance. Specifically, we investigate how variations in context window size and embedding dimensionality influence classification effectiveness when embeddings are used as input features.

The objectives of this study are threefold:

- to systematically evaluate the effect of context window size and vector dimensionality on Word2Vec and GloVe embeddings using an extrinsic classification task;
- to assess whether high-dimensional embeddings reported in prior studies yield measurable performance gains compared to moderate-dimensional alternatives;
- to provide practical guidance for selecting computationally efficient embedding configurations suitable for real-world NLP applications.

The remainder of the paper is structured as follows. Section 2 reviews related work and relevant background on word embedding hyperparameters. Section 3 describes the datasets, preprocessing steps, embedding methods, and classification models used in the experiments. Section 4 presents and analyzes the experimental results. Finally, Section 5 discusses the findings, outlines their implications for research and practice, and concludes the paper.

## 2 BACKGROUND AND RELATED WORK

Word embedding models aim to represent words as dense vectors that encode semantic and syntactic relationships based on distributional properties of language. Among the most influential approaches are predictive models such as Word2Vec and count-based models such as GloVe, both of which have been extensively adopted in a wide range of NLP applications.

The Word2Vec model (Mikolov, Chen, et al., 2013; Mikolov, Sutskever, et al., 2013) is based on the distributional hypothesis, according to which words that appear in similar contexts tend to have similar meanings. Two principal architectures are commonly employed: Continuous Bag-of-Words (CBOW) and Skip-gram. CBOW predicts a target

word from its surrounding context words, whereas Skip-gram predicts surrounding context words from a given target word. While CBOW is generally considered more efficient and performs well for frequent words, Skip-gram has been shown to better capture representations of infrequent words (Mikolov, Chen, et al., 2013).

Two hyperparameters play a central role in determining the quality of Word2Vec embeddings: the context window size and the dimensionality of the embedding space. The window size defines the number of neighboring words considered during training and directly influences the type of linguistic information captured. Levy and Goldberg (2014) argue that smaller windows tend to emphasize syntactic relationships, while larger windows capture broader semantic associations. The dimensionality of the embedding vectors determines the capacity of the model to encode linguistic regularities. While low-dimensional embeddings may lack expressive power, excessively high dimensionalities can introduce redundancy and increase computational cost without providing proportional performance gains. Commonly reported dimensionality values in the literature typically range from 50 to 500 (Levy and Goldberg, 2014; Mikolov, Chen, et al., 2013).

In contrast to predictive models, GloVe (Global Vectors) (Pennington et al., 2014) employs a count-based approach based on matrix factorization of global word co-occurrence statistics. The model constructs a word–word co-occurrence matrix and learns vector representations such that the dot product of two word vectors approximates the logarithm of their co-occurrence probability. This formulation allows GloVe to effectively capture linear regularities in the embedding space and has been shown to perform well on intrinsic evaluation tasks such as word analogy and similarity benchmarks.

Several studies have examined the influence of window size and dimensionality on embedding quality, although their conclusions vary substantially. Adewumi et al. (2020) conducted an extensive analysis across a wide range of dimensionalities (100–3000) and window sizes, concluding that optimal configurations are task-dependent and that increasing dimensionality beyond a certain threshold may even degrade performance. Yang et al. (2018), focusing on a Twitter election classification task, reported optimal results using a dimensionality of 800 and a window size of 10, highlighting the importance of domain alignment between the embedding corpus and the downstream task. Nazir et al. (2022) investigated Word2Vec embeddings for a low-resourced language and found that higher dimensionalities (up to 500) performed best under intrinsic evaluation using word similarity benchmarks.

Beyond performance, embedding stability has also been explored. Chugh et al. (2018) analyzed the stability of Word2Vec embeddings across varying dimensionalities and word frequency bands, demonstrating that dimensionality significantly affects embedding consistency and that stability is corpus-dependent. Their findings further suggest that embeddings for high-frequency words are generally more stable than those for medium- and low-frequency words.

Although these studies provide valuable insights, they differ substantially in terms of embedding models, hyperparameter ranges, evaluation methodologies, and downstream tasks. As summarized in Table 1, prior work reports markedly different recommendations for both context window size and embedding dimensionality, ranging from moderate configurations to very high-dimensional settings exceeding 800 dimensions.

***Table 1.*** *Overview of prior studies investigating Word2Vec and GloVe hyperparameters with respect to window size, dimensionality, and evaluation methodology.*

| Study | Embedding Model | Window Size(s) | Dimensionality Range | Evaluation Type | Downstream Task |
|---|---|---|---|---|---|
| Mikolov et al. (2013) | Word2Vec | Variable | 50–300 | Intrinsic | Analogies |
| Adewumi et al. (2020) | Word2Vec | 4, 8 | 100–3000 | Extrinsic | Multiple NLP tasks |
| Yang et al. (2018) | Word2Vec | 10 | 800 | Extrinsic | Twitter classification |
| Nazir et al. (2022) | Word2Vec | 3, 5, 7 | 100–500 | Intrinsic | Word similarity |
| **This study** | Word2Vec, GloVe | 4–8 | 20–300 | Extrinsic | Spam classification |

Notably, many of these recommendations are derived either from intrinsic evaluation metrics or from task-specific experimental setups, which limits their generalizability. As a consequence, the practical necessity of high-dimensional embeddings for downstream classification tasks remains insufficiently validated under controlled and

reproducible conditions. This study directly addresses this gap by providing a systematic extrinsic evaluation of commonly used Word2Vec and GloVe hyperparameters within a unified experimental framework.

This study addresses this gap by systematically validating the effects of context window size and embedding dimensionality for Word2Vec and GloVe models using an extrinsic evaluation framework based on classification performance. Rather than proposing new embedding techniques, the focus is placed on empirically assessing whether commonly reported hyperparameter recommendations yield measurable benefits in realistic NLP classification scenarios.

## 3 MATERIALS AND METHODS

To systematically evaluate the impact of context window size and embedding dimensionality on word embedding effectiveness, a controlled experimental framework was adopted. The overall workflow of the study is summarized in Fig. 1 and consists of the following stages:

1. Text corpus preprocessing: The input corpus was normalized and tokenized to obtain a representation suitable for training word embedding models.
2. Training of word embedding models: Multiple embedding configurations were generated by systematically varying the embedding method, context window size, and vector dimensionality.
3. Feature vector construction: Each trained embedding model was used to transform text instances into numerical feature vectors for a downstream classification task.
4. Classifier training: For each embedding configuration, a corresponding classifier was trained using identical data splits.
5. Performance evaluation: Classification performance was evaluated using standard metrics, including accuracy, precision, recall, and F1-score.
6. Result analysis: The results were analyzed to identify trends attributable to variations in embedding hyperparameters.

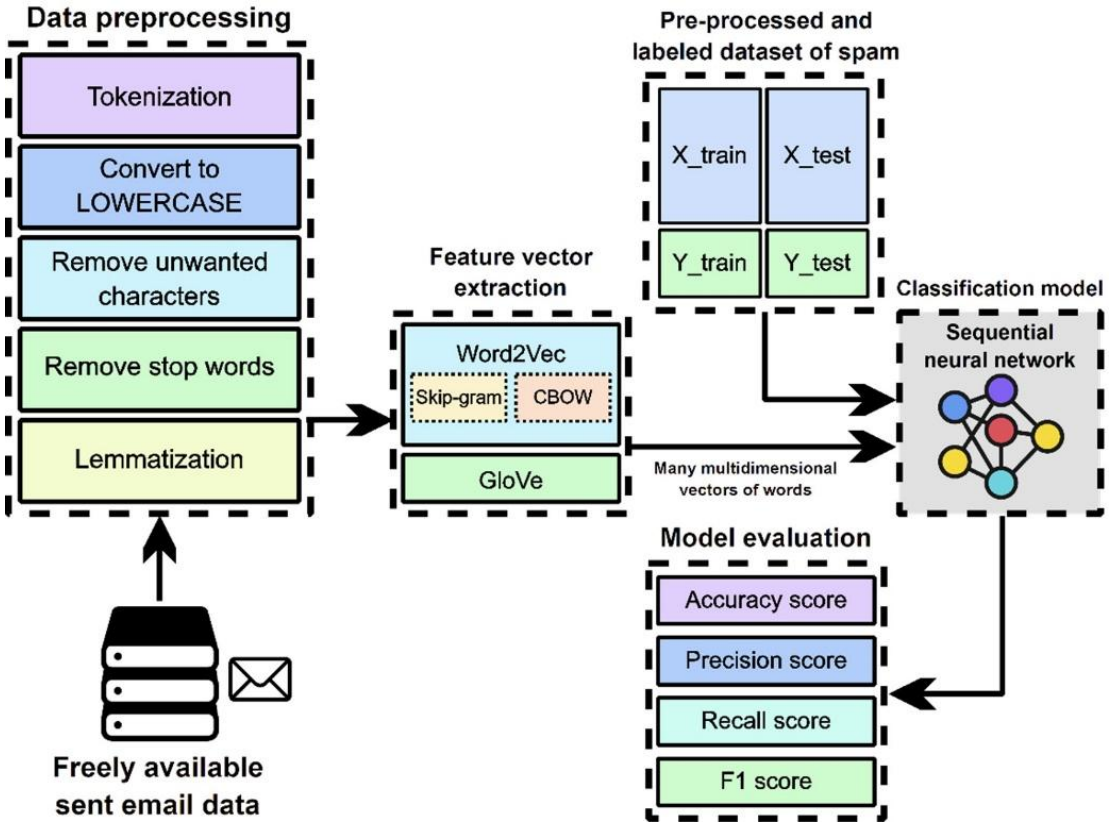


***Figure 1.*** *Methodological steps.*

This baseline experimental design was constructed to isolate the influence of embedding hyperparameters while keeping all other components of the pipeline fixed. In total, 225 embedding–classifier configurations were evaluated in the baseline study, using the same dataset, preprocessing pipeline, and evaluation protocol to ensure comparability across experiments.

The effectiveness of the embedding configurations was assessed using an extrinsic evaluation framework based on supervised text classification. Rather than evaluating embeddings in isolation, their quality was judged by their contribution to downstream classification performance, reflecting their practical use in real-world NLP applications.

In addition to the baseline experiments, supplementary analyses were conducted to further examine the robustness of observed trends for selected embedding models. These supplementary experiments follow the same preprocessing and evaluation pipeline and are introduced later in the methodology section.

The subsequent subsections describe the datasets and preprocessing steps (Section 3.1), the embedding models and parameter settings (Section 3.2), the classification models and evaluation metrics (Section 3.4), and the supplementary experimental design (Section 3.5).

## 3.1 Datasets and Text Data Pre-processing

Two publicly available datasets were used in this study, each serving a distinct role within the experimental framework. The Enron Email Dataset (Klimt and Yang, 2004) was employed as the source corpus for training word embedding models, while the SMS Spam Collection dataset (Almeida et al., 2011) was used exclusively for downstream classification and evaluation.

The Enron Email Dataset consists of approximately 500,000 English-language emails exchanged among 150 users and is commonly used as a large-scale corpus for NLP research. In this study, the dataset was utilized solely to learn word representations, ensuring that embedding training was performed on a substantially larger and more diverse text corpus than the classification dataset. This design choice reduces the risk of information leakage and better reflects realistic embedding usage scenarios.

For extrinsic evaluation, the SMS Spam Collection dataset was selected due to its widespread use in spam classification research (Abayomi-Alli et al., 2019; Dutta et al., 2023; Waja et al., 2023). The dataset contains 5,566 labeled short text messages, including 747 spam messages and 4,819 non-spam ("ham") messages. These messages served as input instances for evaluating the effectiveness of the generated word embeddings in a binary classification task.

Text preprocessing was applied consistently across both datasets using Python to ensure comparability. The preprocessing pipeline consisted of the following steps: sentence segmentation and tokenization, conversion of all tokens to lowercase, removal of tokens containing numerical characters or non-alphabetic symbols, elimination of stop words, and lemmatization to reduce words to their base forms. Tokenization and stop-word removal were implemented using the Natural Language Toolkit (NLTK), while lemmatization was performed using the WordNet lemmatizer. The same preprocessing pipeline was applied to both the embedding corpus and the classification dataset to maintain consistency across experiments.

This standardized preprocessing procedure ensured that variations in classification performance could be attributed primarily to differences in embedding configurations rather than inconsistencies in text normalization.

## 3.2 Word Embedding Methods

This study focuses on widely used static word embedding techniques, namely Word2Vec and GloVe, due to their continued relevance in practical NLP pipelines and their sensitivity to hyperparameter selection. All embedding models were trained using the same preprocessed corpus to ensure comparability across configurations.

### 3.2.1 Word2Vec

Word2Vec is a predictive embedding model that learns vector representations of words by exploiting local contextual information. Two architectures were considered in this study: Continuous Bag-of-Words (CBOW) and Skip-gram (Mikolov, Chen, et al., 2013; Mikolov, Sutskever, et al., 2013). Rather than providing a detailed

architectural description, which is well documented in the literature, this section focuses on the experimental configuration relevant to reproducibility.

Both CBOW and Skip-gram models were trained using the Gensim library (Řehůřek and Sojka, 2010). The key hyperparameters under investigation were the context window size and the dimensionality of the embedding space. Unless stated otherwise, all other parameters were kept at their default values as defined by the Gensim implementation to minimize confounding effects. Negative sampling was employed during training to improve computational efficiency.

The CBOW architecture predicts a target word from its surrounding context words and is generally more efficient for frequent terms, while the Skip-gram architecture predicts surrounding context words from a given target word and is often better suited for representing infrequent words. Both architectures were evaluated under identical experimental conditions to isolate the effect of hyperparameter variation.

### 3.2.2 GloVe

GloVe (Global Vectors) is a count-based embedding model that learns word representations by factorizing a global word co-occurrence matrix (Pennington et al., 2014). Unlike predictive models, GloVe explicitly leverages global statistical information from the corpus, which has been shown to be effective for capturing linear regularities in embedding spaces.

In this study, GloVe embeddings were trained using the Gensim implementation, with the same preprocessing pipeline and vocabulary as the Word2Vec models. As with Word2Vec, the primary parameters of interest were the context window size used to construct the co-occurrence matrix and the dimensionality of the resulting embeddings. All remaining parameters were retained at their default settings to ensure consistency across experiments.

### 3.2.3 Parameter Scope and Experimental Consistency

Across all embedding methods, the same preprocessing pipeline, training corpus, and vocabulary constraints were applied. The systematic variation of context window size and embedding dimensionality constitutes the core experimental manipulation of the baseline study. This design ensures that observed performance differences can be attributed primarily to embedding hyperparameters rather than implementation-specific factors.

## 3.3 Classification Models

To evaluate the effectiveness of word embeddings in a downstream task, a supervised text classification framework was employed. In the baseline experimental setup, a neural network classifier was used to ensure a consistent and controlled evaluation of embedding configurations across all models.

The baseline classifier was implemented as a sequential neural network using the Keras library. The architecture consisted of an embedding layer initialized with pretrained word vectors, followed by a flattening layer and a fully connected output layer with a sigmoid activation function for binary classification. The embedding layer was configured using the vocabulary size and embedding dimensionality corresponding to each embedding model and was kept non-trainable to isolate the effect of pretrained embeddings.

For Word2Vec embeddings, models were trained using the Gensim implementation with a minimum word frequency threshold of 3, a fixed random seed of 7, and five worker threads. Both CBOW and Skip-gram architectures were evaluated, depending on the experimental configuration. For GloVe embeddings, the co-occurrence matrix was constructed using the same context window size, and embeddings were trained with a learning rate of 0.05 for 30 epochs. In all cases, word vectors were generated exclusively for words appearing in the classification dataset to ensure a consistent vocabulary.

Text messages were converted into sequences of integer indices based on the learned vocabulary. Input sequences were padded to a fixed length of 20 tokens. The neural network was trained using the RMSprop optimizer and binary cross-entropy loss for five epochs. A fixed train–test split and random seed were used across all experiments to ensure reproducibility.

### 3.4 Evaluation Protocol and Performance Metrics

The performance of the classification models was assessed using standard evaluation metrics commonly applied in binary classification tasks: accuracy, precision, recall, and F1-score. These metrics were selected to provide a balanced evaluation of model performance, particularly in the presence of class imbalance.

Accuracy reflects the overall proportion of correctly classified instances, while precision and recall capture complementary aspects of classification quality with respect to false positives and false negatives. The F1-score, defined as the harmonic mean of precision and recall, provides a single aggregated measure that balances both error types.

All evaluation metrics were computed using the Scikit-learn library (Pedregosa et al., 2011). To ensure comparability across experiments, the same evaluation protocol and data splits were applied to all embedding configurations.

### 3.5 Supplementary Classification Models

Based on the results obtained in the baseline experiments, Word2Vec embeddings consistently outperformed GloVe embeddings across the evaluated configurations. In addition, Word2Vec demonstrated more stable behavior with respect to variations in embedding dimensionality and context window size. For these reasons, subsequent supplementary experiments focused exclusively on Word2Vec embeddings.

To further assess the robustness and generalizability of the observed trends, additional experiments were conducted using classical machine learning classifiers. These classifiers represent different learning paradigms and allow the evaluation of embedding quality independently of neural network–specific effects. The evaluated classifiers included Logistic Regression, Support Vector Machines, an SGD-based linear classifier, and Bernoulli Naive Bayes.

All classifiers were implemented using the Scikit-learn library with default parameter settings unless stated otherwise. The same preprocessing pipeline, embedding configurations, data splits, and evaluation metrics as in the baseline experiments were applied to ensure comparability. These supplementary experiments serve to validate whether the conclusions drawn from the neural network–based evaluation persist across alternative classification models.

## 4 RESULTS

This section presents the results of the experimental evaluation described in Section 3. The objective is to assess how variations in context window size and embedding dimensionality influence downstream classification performance. Results are first reported for the baseline neural network classifier and subsequently extended through additional analyses.

### 4.1 Baseline Neural Network Results

Baseline experiments were conducted using a fixed neural network classifier to ensure a controlled comparison of embedding configurations. All results reported in this subsection are based on identical preprocessing, training–testing splits, and network architecture, such that observed differences can be attributed primarily to the properties of the word embeddings.

Across all evaluated configurations, Word2Vec embeddings consistently outperformed GloVe embeddings in terms of median classification performance. Among Word2Vec variants, the Skip-gram architecture achieved the highest overall scores across most evaluation metrics, while CBOW exhibited more homogeneous behavior with lower variance. These findings motivated a more detailed analysis of individual embedding hyperparameters, which is presented in the following subsections.

### 4.2 Effect of Context Window Size and Embedding Dimensionality

#### 4.2.1 Context Window Size

The effect of context window size on classification performance is illustrated in Fig. 2–3. For the Skip-gram architecture, the highest median accuracy was observed at a window size of 5 (median accuracy = 0.9449). Window

sizes of 5 and 6 also yielded the highest upper quartile and maximum accuracy values, whereas performance differences between window sizes were relatively small.

In contrast, CBOW and GloVe embeddings exhibited their highest median accuracy values at a window size of 7 (0.9410 and 0.9304, respectively). Similar trends were observed for precision and recall metrics. For Skip-gram, precision and recall were maximized at smaller window sizes, while CBOW demonstrated improved recall at larger windows.

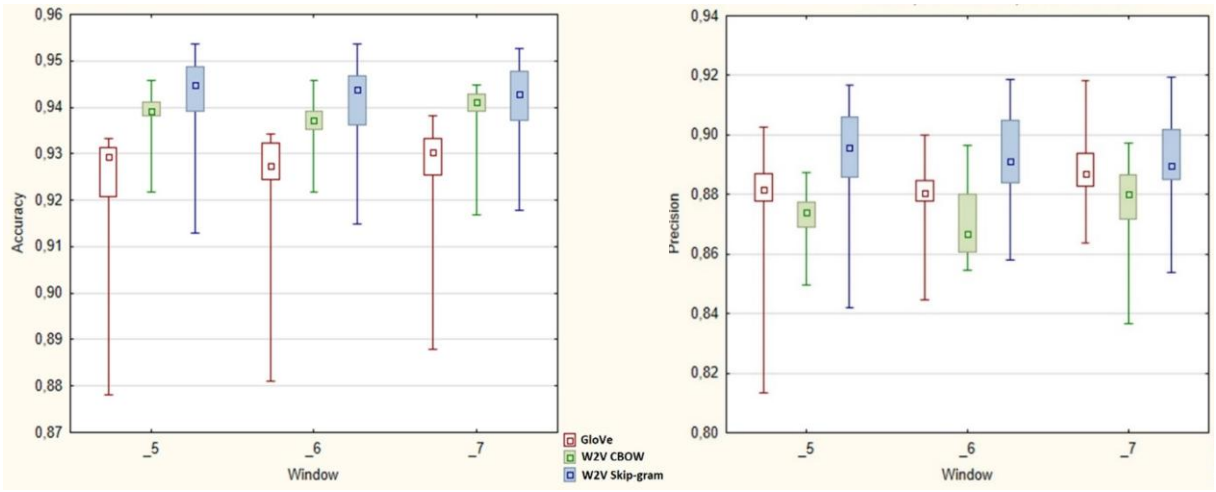


***Figure 2.*** *Boxplot of Accuracy (left) and Precision (right) metrics for window sizes.*

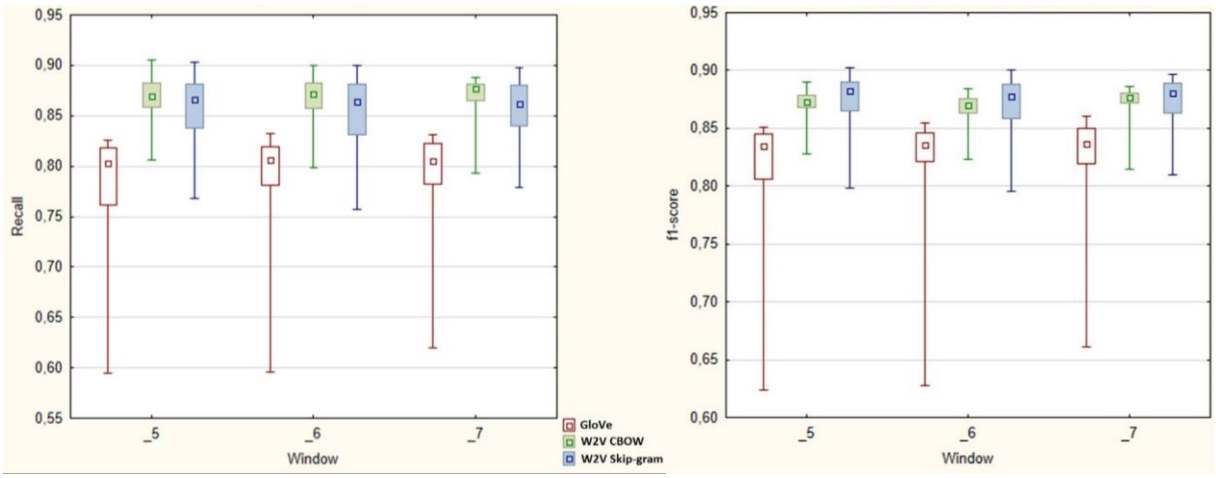


***Figure 3.*** *Boxplot of Recall (left) and F1-score (right) metrics for window sizes.*

Overall, the results indicate that the optimal window size is dependent on the embedding method, although differences between window sizes remain moderate. Due to the limited effect sizes and overlapping distributions, no formal statistical significance testing was conducted at this stage.

### 4.2.2 Embedding Dimensionality

The effect of embedding dimensionality on classification performance was analyzed using the F1-score as the primary evaluation metric. Figures 4–6 illustrate the variation of F1-score across different embedding dimensionalities for context window sizes of 5, 6, and 7, respectively.

Across all embedding methods, dimensionalities below approximately 50 resulted in substantially lower F1-score values. The most pronounced performance improvements were observed up to around 100 dimensions. Beyond this point, performance gains diminished, and F1-score values remained relatively stable across larger dimensionalities.

For the Word2Vec Skip-gram architecture, consistently high and stable F1-score values were achieved for dimensionalities exceeding approximately 100, with no systematic improvements observed beyond roughly 150 dimensions. The CBOW architecture exhibited a comparable trend, although peak performance was reached at slightly higher dimensionalities. In contrast, GloVe embeddings showed greater variability across dimensionality settings and generally lower F1-score values.

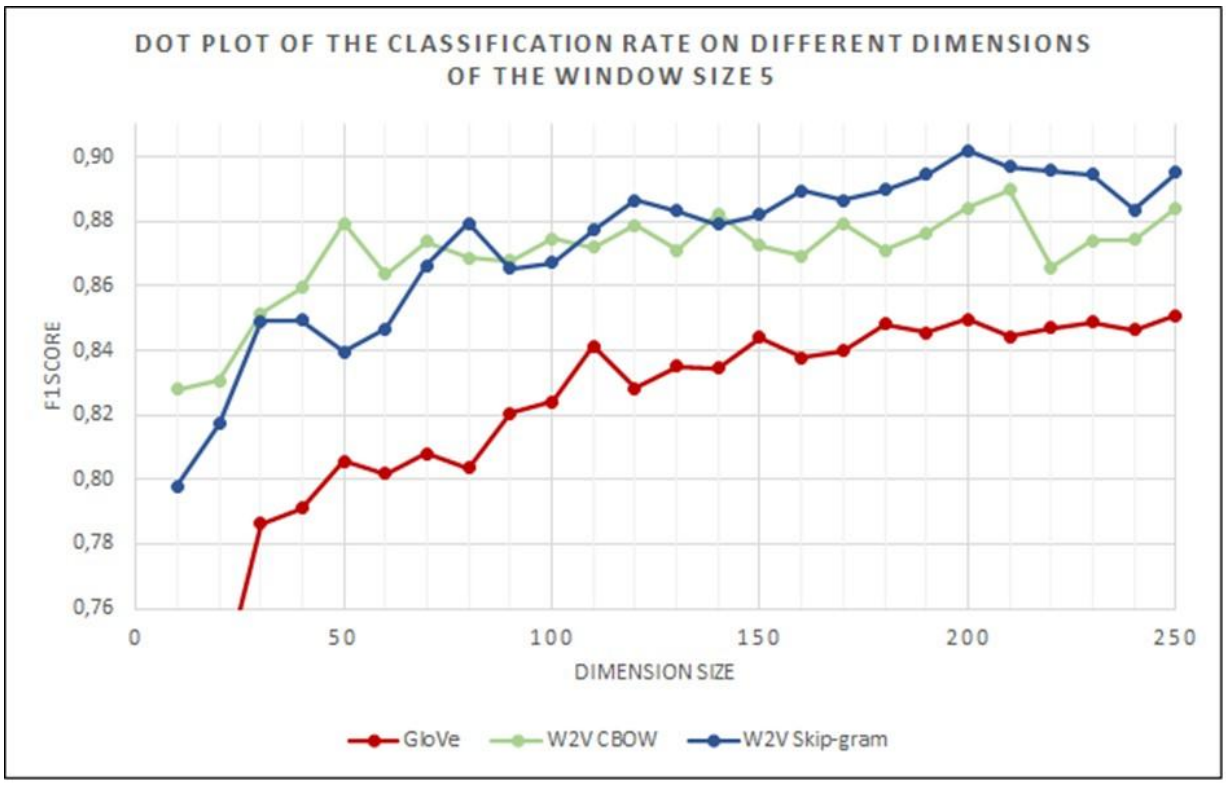


***Figure 4.*** *Dot plots of F1-score metric on different dimensions of the window size 5.*

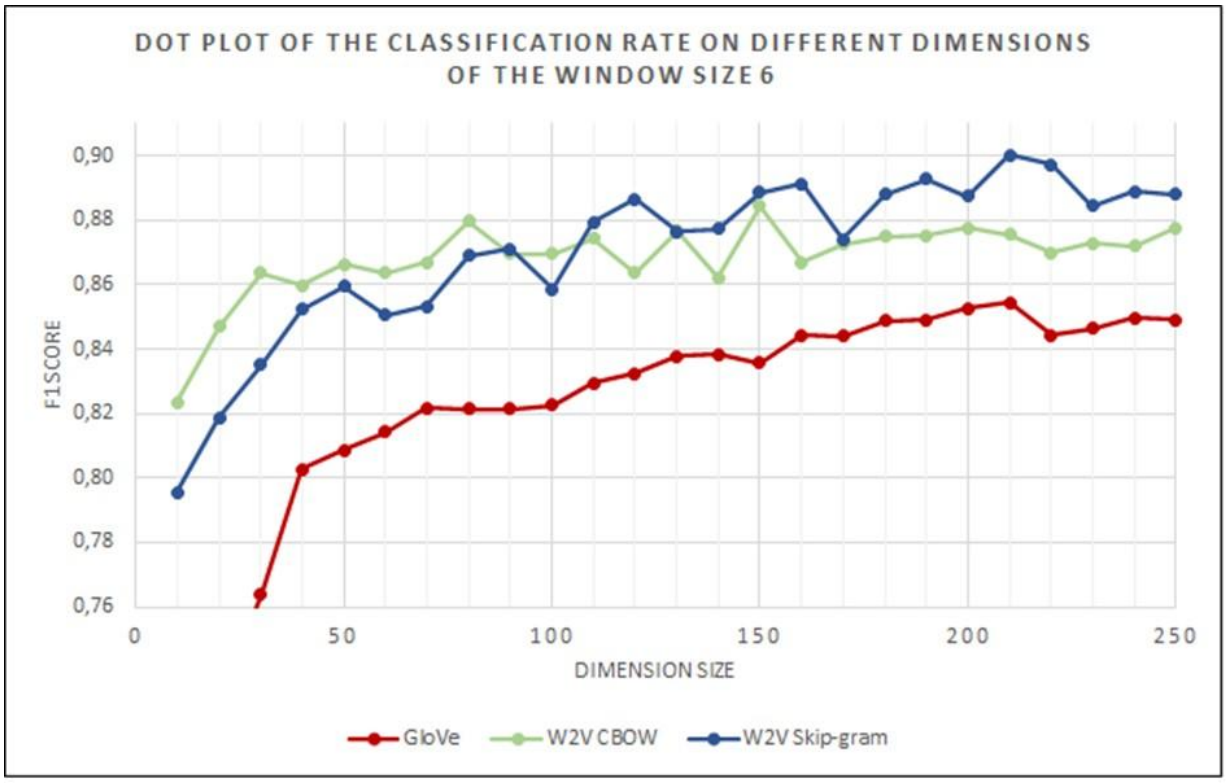


***Figure 5.*** *Dot plots of F1-score metric on different dimensions of the window size 6.*

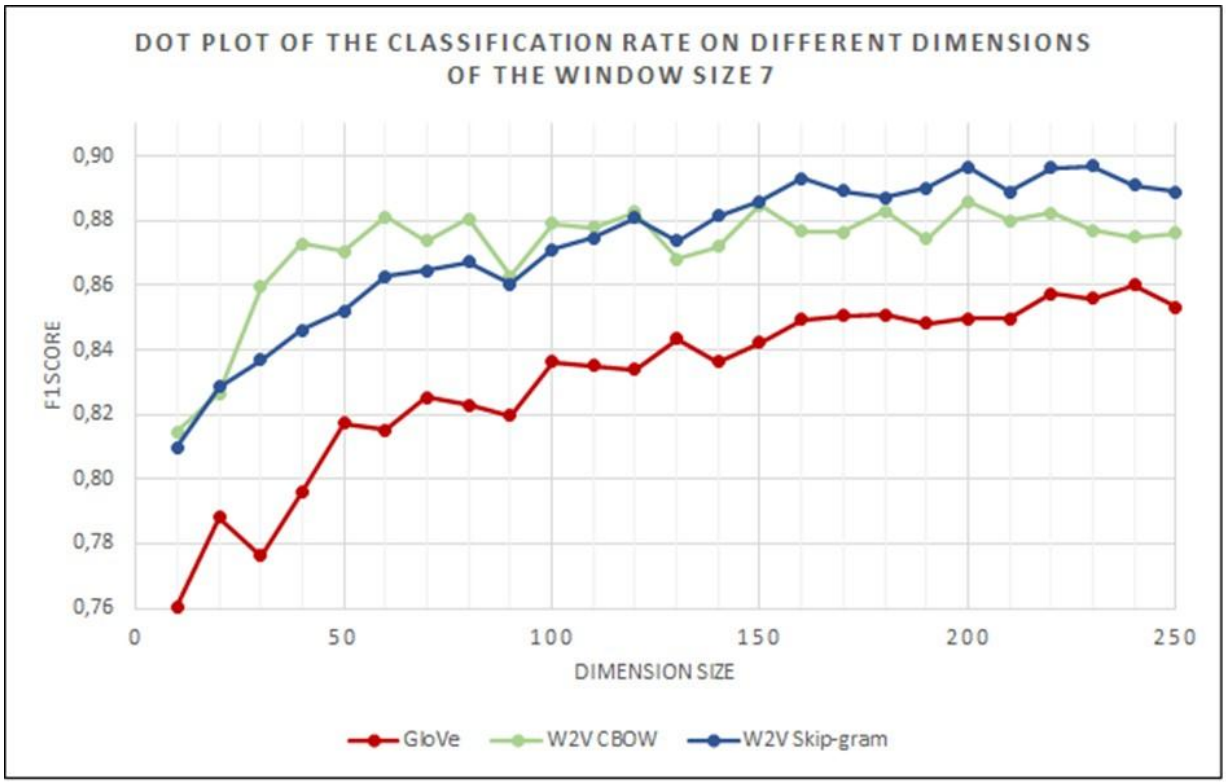


***Figure 6.*** *Dot plots of F1-score metric on different dimensions of the window size 7.*

Overall, the results indicate that increasing embedding dimensionality beyond moderate values yields only marginal benefits in terms of F1-score performance. Embedding dimensionalities in the range of approximately 100–150 appear sufficient to achieve stable and competitive classification performance for the evaluated task.

## 4.3 Classical Machine Learning Classifiers

Based on the baseline evaluation, Word2Vec embeddings demonstrated superior and more stable performance compared to GloVe. Consequently, extended experiments were conducted exclusively using Word2Vec embeddings to assess the robustness of the observed trends across a broader range of hyperparameter settings and classification models.

In this supplementary analysis, both CBOW and Skip-gram architectures were evaluated. Embedding dimensionality was varied from 20 to 300 in steps of 20, and context window sizes ranged from 4 to 8. Classification performance was assessed using four classical machine learning classifiers: Logistic Regression, Support Vector Machines, SGD-based linear classifiers, and Bernoulli Naive Bayes. Performance was measured using the F1-score, computed as the arithmetic mean of class-wise F1 values.

### 4.3.1 CBOW Embeddings

The results for CBOW embeddings indicate relatively stable classification performance across a wide range of hyperparameter configurations. As shown in Fig. 7, which depicts the average F1-score as a function of embedding dimensionality for individual classifiers, CBOW embeddings exhibit only weak dependence on dimensionality. While slightly higher F1-scores are occasionally achieved at larger dimensionalities, performance differences between dimensions above approximately 200 remain marginal.

The influence of context window size is illustrated in Fig. 8, which presents the average F1-score across all dimensionalities for each window size. Smaller context windows consistently yield higher F1-scores, with window size 4 achieving the best overall performance. Increasing the window size beyond this value results in a gradual decrease in average F1-score.

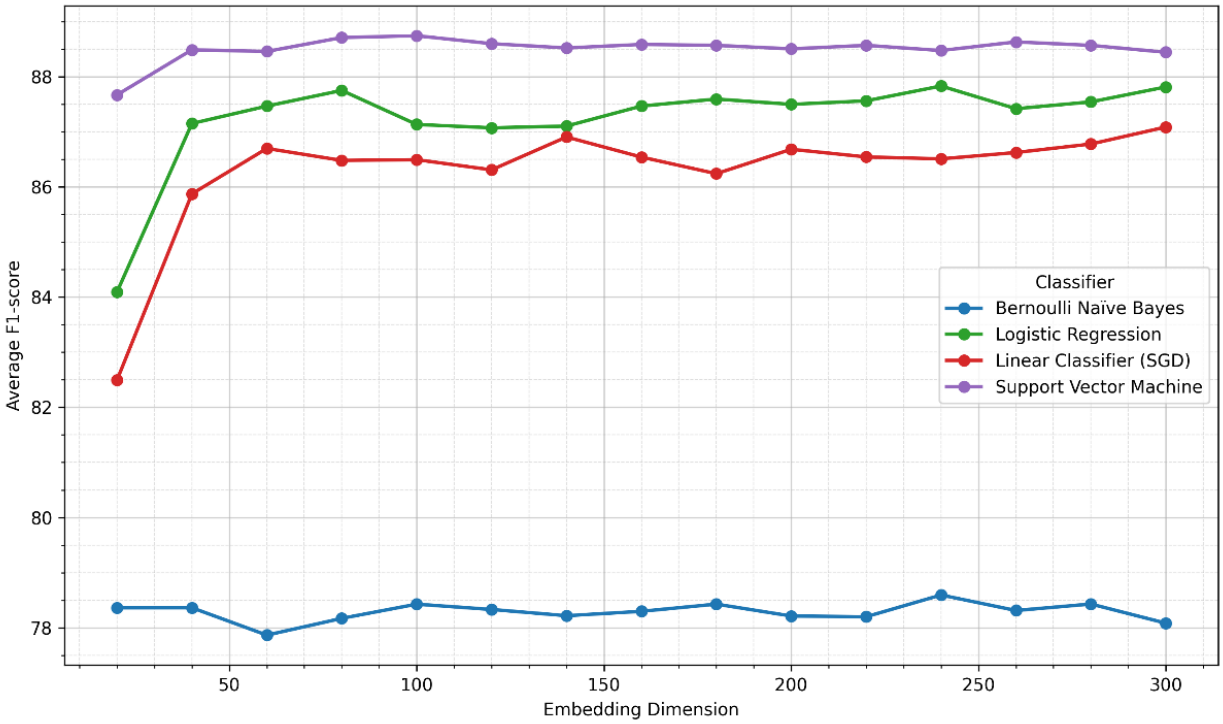


***Figure 7.*** *CBOW: F1 vs. dimension.*

This behavior is likely influenced by the relatively limited contextual complexity of the dataset. In settings where semantic dependencies are predominantly local, smaller context windows may provide sufficiently informative representations without introducing additional noise from broader contextual aggregation.

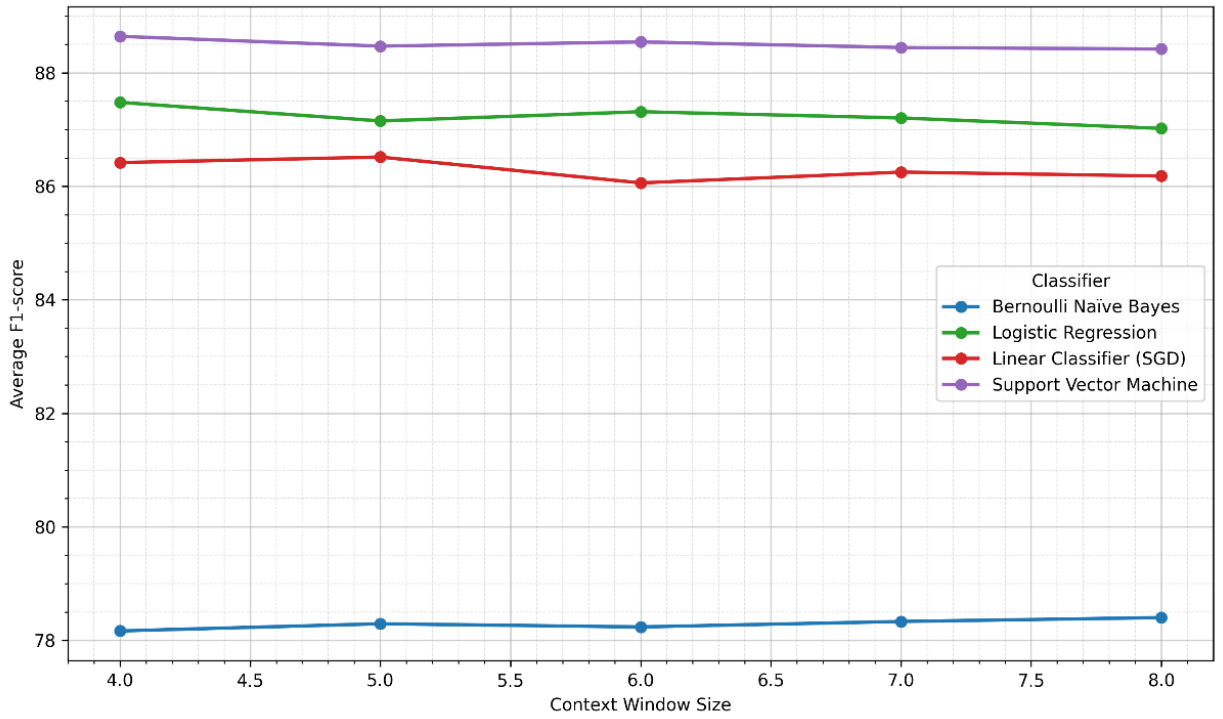


***Figure 8.*** *CBOW: F1 vs. window size.*

Classifier choice has a more pronounced impact on performance than either embedding dimensionality or window size. Support Vector Machines and Logistic Regression consistently achieve the highest F1-scores, frequently exceeding 87%, whereas Bernoulli Naive Bayes exhibits lower overall performance. These trends are consistent

across both dimensionality and window size variations, indicating that CBOW embeddings are particularly well suited for linear and margin-based classifiers.

### 4.3.2 Skip-gram Embeddings

Skip-gram embeddings achieve consistently high classification performance across the examined hyperparameter configurations. As shown in Fig. 9, which illustrates the relationship between embedding dimensionality and average F1-score for individual classifiers, performance remains relatively stable across dimensionalities from 20 to 300. While moderate improvements are observed when increasing dimensionality from very low values (20–60), performance tends to saturate beyond approximately 100 dimensions, with only marginal variations thereafter.

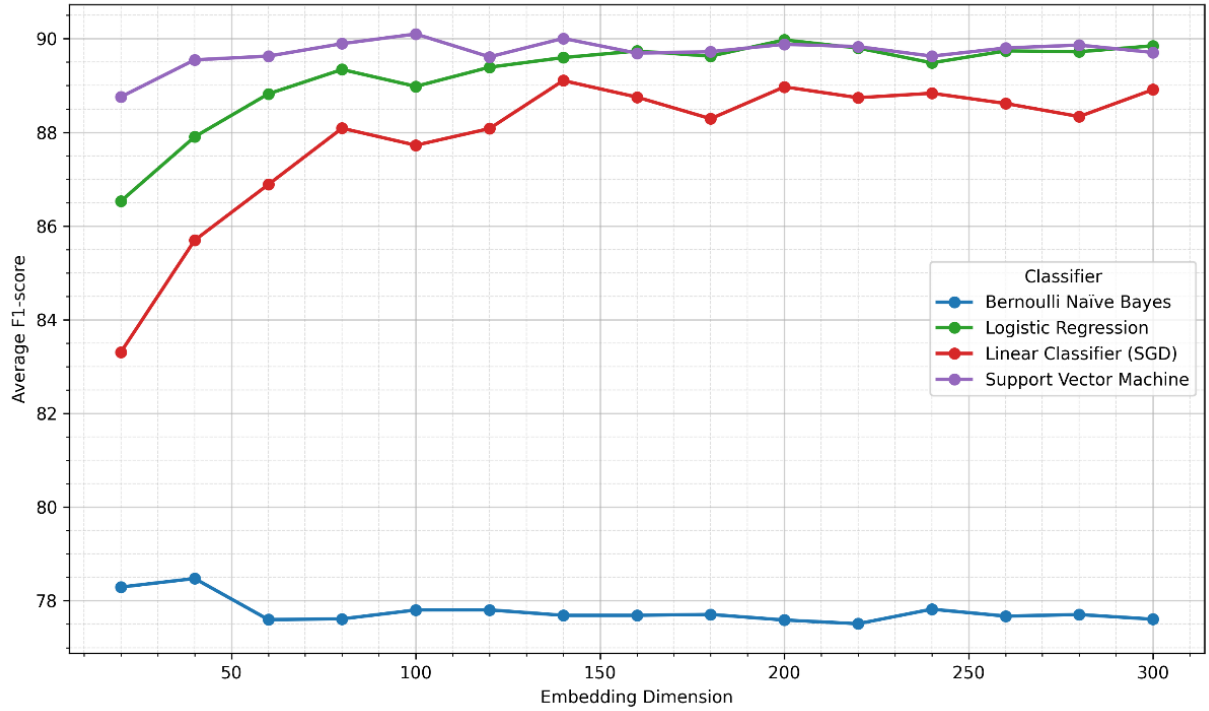


***Figure 9.*** *Skip-gram: F1 vs. dimension.*

The influence of context window size is presented in Fig. 10. Smaller window sizes, particularly window size 4, yield slightly higher F1-scores across all classifiers; however, the overall differences between window sizes 4–8 remain small. This suggests that, similarly to CBOW, local contextual information is sufficient for effective representation learning in the examined dataset.

Classifier-dependent differences are clearly observable. Support Vector Machines consistently achieve the highest F1-scores (approximately 90%), closely followed by Logistic Regression. The SGD-based linear classifier yields slightly lower but still competitive performance. In contrast, Bernoulli Naive Bayes performs substantially worse and exhibits limited sensitivity to both dimensionality and window size. Overall, the results indicate that Skip-gram embeddings, similar to CBOW, are particularly well aligned with linear and margin-based classification models.

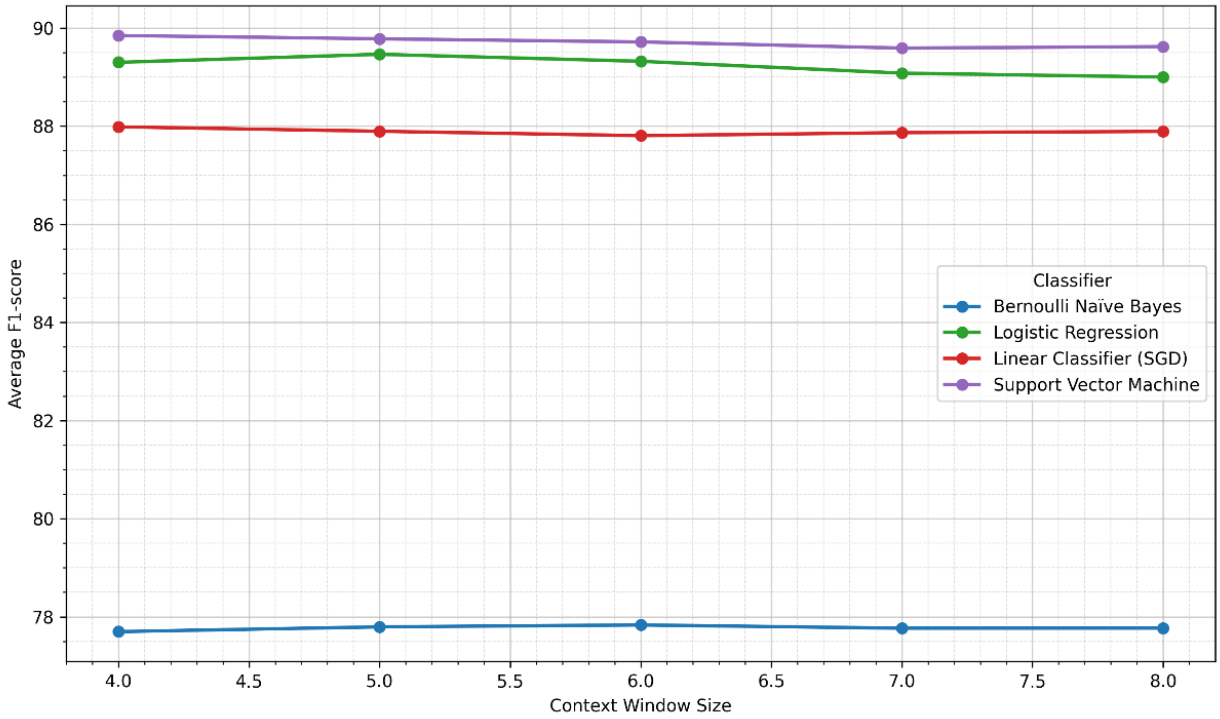


***Figure 10.*** *Skip-gram: F1 vs. window size.*

## 5 DISCUSSION

This study investigated the influence of context window size and embedding dimensionality on the effectiveness of Word2Vec and GloVe embeddings in a downstream text classification task. Rather than proposing a new embedding technique, the primary objective was to provide a systematic extrinsic validation of commonly used hyperparameter settings under a controlled and reproducible experimental framework.

The results confirm that both context window size and embedding dimensionality influence classification performance, although their effects differ in magnitude and consistency. With respect to context window size, the observed differences between configurations were generally moderate. For individual embedding methods, specific window sizes yielded slightly better performance; however, performance distributions largely overlapped across window sizes. This suggests that while window size selection matters, it does not constitute a dominant factor in downstream classification performance within the evaluated range.

The choice to evaluate a limited set of window sizes was motivated by the aim to maintain a controlled experimental design focused on the most commonly used configurations reported in prior studies. Importantly, the results indicate that expanding the window size beyond this range would be unlikely to yield substantial performance gains, as trends remained stable and differences were small. This observation is further supported by the extended Word2Vec experiments, where smaller window sizes consistently achieved competitive or superior performance across multiple classifiers.

Embedding dimensionality exhibited a clearer and more systematic effect. Very low-dimensional embeddings resulted in degraded performance, whereas increasing dimensionality beyond moderate values led to diminishing returns. Across both baseline and extended experiments, stable classification performance was achieved using moderate-dimensional embeddings, confirming that excessively high-dimensional representations are not necessary for effective downstream classification.

These findings align with and extend prior observations reported by Adewumi et al. (2020), who emphasized task-specific optimal configurations and noted performance degradation at very high dimensionalities. Similarly, while Yang et al. (2018) reported strong performance using substantially higher dimensionalities and larger context windows, our results suggest that comparable classification performance can be achieved with considerably lower parameter values when embeddings are evaluated extrinsically.

From a practical perspective, the results provide guidance for selecting embedding hyperparameters in text classification tasks. The experiments indicate that moderate-dimensional Word2Vec embeddings offer a favorable balance between performance stability and computational efficiency. Increasing dimensionality beyond this range leads to minimal performance improvements while incurring additional computational cost.

The extended evaluation across multiple classifiers further highlights that embedding effectiveness is strongly influenced by the downstream model. Linear and margin-based classifiers, such as Logistic Regression and Support Vector Machines, consistently achieved the highest and most stable performance across embedding configurations. In contrast, Bernoulli Naive Bayes yielded substantially lower performance and demonstrated limited benefit from increasing embedding dimensionality. These observations reinforce the importance of jointly considering embedding configuration and classifier characteristics when designing text classification pipelines.

An important aspect of the findings is the relatively weak dependence of performance on the examined hyperparameters within the tested ranges. While this may limit the extent of novel performance gains achieved through hyperparameter tuning alone, it also indicates a degree of robustness in Word2Vec-based representations for the studied task. In practical settings, such robustness may be advantageous, as it reduces the need for extensive parameter optimization and allows reliable performance to be achieved with moderate embedding dimensionalities and small context windows.

Nevertheless, several limitations should be acknowledged. First, the study was conducted using a single downstream classification task and dataset. The relatively limited contextual complexity of the dataset may partially explain why smaller context windows consistently achieved optimal or near-optimal performance. In tasks involving longer-range semantic dependencies or more complex discourse structures, larger context windows or higher-dimensional embeddings might yield different outcomes. Therefore, the generalizability of the observed trends to other domains and tasks should be interpreted with caution.

Second, although the experiments systematically varied embedding dimensionality and context window size, the evaluated hyperparameter space remains constrained to commonly used configurations. While this design choice ensured comparability and reproducibility, alternative parameter ranges or training regimes could produce different performance dynamics.

Finally, the study focused exclusively on static word embeddings (Word2Vec and GloVe) evaluated in an extrinsic classification setting. Contextualized embedding models, such as transformer-based representations, were not considered. Future research could extend this comparative framework to dynamic embeddings and more diverse downstream tasks in order to further assess the robustness and transferability of hyperparameter effects.

## 6 CONCLUSION

This study presented an empirical extrinsic evaluation of context window size and embedding dimensionality for static word embedding models in a text classification task. Word2Vec (CBOW and Skip-gram) and GloVe embeddings were evaluated using a unified experimental pipeline and downstream spam classification performance.

In the baseline experiments, Word2Vec embeddings consistently outperformed GloVe embeddings, achieving higher and more stable classification performance across all evaluated configurations. Across the evaluated window sizes, performance differences were generally moderate, although consistent trends were observed. Word2Vec Skip-gram achieved its best performance at smaller context windows (around 5), whereas Word2Vec CBOW and GloVe achieved their highest median performance at slightly larger window sizes (around 7). Overall, differences between window sizes remained relatively small, indicating limited sensitivity within the evaluated range.

Embedding dimensionality exhibited a clearer impact on classification performance. Very low-dimensional embeddings (below approximately 50 dimensions) consistently resulted in degraded performance, whereas increasing dimensionality beyond moderate values led to diminishing returns. Stable and competitive performance was typically achieved using embedding dimensionalities in the range of approximately 100–150, with no systematic improvements observed for higher dimensions. These trends were consistent across both baseline and extended experiments.

The extended evaluation using classical machine learning classifiers confirmed the robustness of these findings. Support Vector Machines and Logistic Regression consistently achieved the highest F1-scores, typically around 0.89–0.90 for Skip-gram embeddings under favorable configurations. In contrast, Bernoulli Naive Bayes exhibited substantially lower and more variable performance across configurations. Across classifiers, smaller context windows (4–5) and moderate embedding dimensionalities yielded the most stable results.

Overall, the results demonstrate that effective classification performance can be achieved without resorting to very large context windows or high-dimensional embeddings. From a practical standpoint, moderate embedding dimensionalities and smaller context windows provide a favorable balance between performance and computational efficiency. While the conclusions are specific to the examined task and dataset, the findings reinforce that optimal embedding hyperparameters are task- and classifier-dependent and should be selected based on downstream performance rather than intrinsic evaluation alone.

## ADDITIONAL INFORMATION AND DECLARATIONS

**Funding:** This work was supported by the Scientific Grant Agency of the Ministry of Education of the Slovak Republic and Slovak Academy of Sciences under Contract VEGA-1/0821/21, also by the Slovak Research and Development Agency under the contract no. APVV-18-0473.

**Conflict of Interests:** The authors declare no conflict of interest

**Author Contributions:** D.D.: Conceptualization, Methodology, Investigation, Software, Data curation, Formal analysis, Writing – Original draft preparation, Writing – Reviewing and editing. J.K.: Investigation, Software, Writing – Original draft preparation, Writing – Reviewing and editing.

**Statement on the Use of Artificial Intelligence Tools:** The authors declare that they didn't use artificial intelligence tools for text or other media generation in this article.

**Data Availability:** The data that support the findings of this study are available from the corresponding author.

## REFERENCES

**Abayomi-Alli, O., Misra, S., Abayomi-Alli, A., & Odusami, M.** (2019). A review of soft techniques for SMS spam classification: Methods, approaches and applications. *Engineering Applications of Artificial Intelligence*, 86, 197–212. https://doi.org/10.1016/j.engappai.2019.08.024

**Abubakar, H. D., Umar, M., & Bakale, M.A.** (2022). Sentiment Classification: Review of Text Vectorization Methods: Bag of Words, Tf-Idf, Word2vec and Doc2vec. *SLU Journal of Science and Technology*, 4(1–2), 27–33. https://doi.org/10.56471/slujst.v4i.266

**Adewumi, T. P., Liwicki, F., & Liwicki, M.** (2020). Word2Vec: Optimal Hyper-Parameters and Their Impact on NLP Downstream Tasks. arXiv:2003.11645. https://doi.org/10.48550/arXiv.2003.11645

**Almeida, T. A., Hidalgo, J. M. G., & Yamakami, A.** (2011). Contributions to the study of SMS spam filtering. In *Proceedings of the 11th ACM Symposium on Document Engineering*, (pp. 259–262). ACM. https://doi.org/10.1145/2034691.2034742

**Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Amodei, D.** (2020). Language Models are Few-Shot Learners. In *34th Conference on Neural Information Processing Systems (NeurIPS 2020)*. NeurIPS.

**Chugh, M., Whigham, P. A., & Dick, G.** (2018). Stability of Word Embeddings Using Word2Vec. In *AI 2018: Advances in Artificial Intelligence* (pp. 812–818). Springer. https://doi.org/10.1007/978-3-030-03991-2_73

**Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K.** (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*, (pp. 4171–4186). ACL. https://doi.org/10.18653/v1/N19-1423

**Dharma, E. M., Lumban Gaol, F., Leslie, H., Warnars, H. S., & Soewito, B**. (2022). The Accuracy Comparison Among word2vec, Glove, and Fasttext Towards Convolution Neural Network (CNN) Text Classification. *Journal of Theoretical and Applied Information Technology*, 100(2), 349–359.

**Dutta, S., Das, A. K., Ghosh, S., & Samanta, D.** (2023). Attribute selection to improve spam classification. In *Data Analytics for Social Microblogging Platforms* (pp. 95–127). Elsevier. https://doi.org/10.1016/B978-0-32-391785-8.00016-0

**Khurana, D., Koli, A., Khatter, K., & Singh, S.** (2023). Natural language processing: state of the art, current trends and challenges. *Multimedia Tools and Applications*, 82(3), 3713–3744. https://doi.org/10.1007/s11042-022-13428-4

**Klimt, B., & Yang, Y.** (2004). The Enron Corpus: A New Dataset for Email Classification Research. In *Machine Learning: ECML 2004* (pp. 217–226). Springer. https://doi.org/10.1007/978-3-540-30115-8_22

**Levy, O., & Goldberg, Y.** (2014). Dependency-Based Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, (pp. 302–308). ACL. https://doi.org/10.3115/v1/P14-2050

**Mikolov, T., Chen, K., Corrado, G., & Dean, J.** (2013). Efficient Estimation of Word Representations in Vector Space. In *1st International Conference on Learning Representations, ICLR 2013*. ICLR.

**Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J.** (2013). Distributed Representations of Words and Phrases and their Compositionality. In *NIPS'13: Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, (pp. 3111–3119). NISP.

**Nazir, S., Asif, M., Sahi, S. A., Ahmad, S., Ghadi, Y. Y., & Aziz, M. H.** (2022). Toward the Development of Large-Scale Word Embedding for Low-Resourced Language. IEEE Access, 10, 54091–54097. https://doi.org/10.1109/ACCESS.2022.3173259

**Pedregosa, F., Varoquaux, G., Gramfort, A., Michel V. and Thirion, B., Grisel, O., Blondel, M., Prettenhofer P. and Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E.** (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

**Pennington, J., Socher, R., & Manning, C.** (2014). Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, (pp. 1532–1543). ACL. https://doi.org/10.3115/v1/D14-1162

**Řehůřek, R., & Sojka, P.** (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New for NLP Frameworks*, (pp. 45–50). University of Malta.

**Waja, G., Patil, G., Mehta, C., & Patil, S.** (2023). How AI Can be Used for Governance of Messaging Services: A Study on Spam Classification Leveraging Multi-Channel Convolutional Neural Network. *International Journal of Information Management Data Insights*, 3(1), 100147. https://doi.org/10.1016/j.jjimei.2022.100147

**Wyatt, J. M., Booth, G. J., & Goldman, A. H.** (2021). Natural Language Processing and Its Use in Orthopaedic Research. *Current Reviews in Musculoskeletal Medicine*, 14(6), 392–396. https://doi.org/10.1007/s12178-021-09734-3

**Yang, X., Macdonald, C., & Ounis, I.** (2018). Using word embeddings in Twitter election classification. *Information Retrieval Journal*, 21(2–3), 183–207. https://doi.org/10.1007/s10791-017-9319-5

*Acta Informatica Pragensia* is published by the Prague University of Economics and Business, Czech Republic | eISSN: 1805-4951