IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

A novel approach for mitigating class imbalance in Arabic text classification

EMAD NABIL[1], ABDELRAHMAN E. NAGIB[2], MENA HANY[3], SAFIULLAH FAIZULLAH[4], and WAEL HASSAN GOMAA[5]

[1]Faculty of Computer and Information Systems, Islamic University of Madinah, Madinah, KSA (e-mail: e.nabil@fci-cu.edu.eg, emadnabil@iu.edu.sa)
[2]Faculty of Computer Science, October University for Modern Sciences and Arts (MSA), Giza, Egypt (e-mail: abezzeldin@msa.edu.eg)
[3]King Fahd University of Petroleum & Minerals, KSA (e-mail: g202411920@kfupm.edu.sa)
[4]Faculty of Computer and Information Systems, Islamic University of Madinah, Madinah, KSA (e-mail: safi@iu.edu.sa)
[5]Faculty of Computers and Artificial Intelligence, Beni-Suef University, Beni-Suef, Egypt (e-mail: wael.goma@gmail.com)
Corresponding author: Emad Nabil (e-mail: e.nabil@fci-cu.edu.eg, emadnabil@iu.edu.sa).

ABSTRACT: Natural language processing (NLP) has become somewhat well-known because of its many uses; deep neural networks have driven major developments. Still, there are difficulties, especially in Arabic NLP, where the language's large vocabulary of over 12 million words and several dialects cause special issues. Arabic has a large speaker base; however, NLP studies in this language find challenges, particularly with class imbalance. Many times, standard class balancing methods overlook intra-class similarity, a crucial element influencing model training. We present a new approach for computing intra-class similarity using cosine similarity and embedding models to find ideal class weights for model training, hence bridging this difference. On two benchmark datasets—the Arabic Semantic Question Similarity dataset (NSURL) and the Microsoft Research Paragraph Corpus (MRPC)—we assessed our method. With an accuracy of state-of-the-art 83.25% on the MRPC dataset and 96.931% on the NSURL dataset, our approach proved successful in improving model performance in Arabic text classification.

INDEX TERMS: Natural Language Processing; Intra-class similarity; Transformers; Classification; Class weights; semantic similarity.

I. INTRODUCTION

Arabic Natural Language Processing (NLP) is still a crucial field of research in computational linguistics because of the unique qualities of the language and its great worldwide reach. With around 400 million speakers worldwide, Arabic is among the most widely used languages; it is also one of the six official languages of the United Nations. This emphasizes the need to create sophisticated NLP methods able to effectively handle its complexity [1].

Every language presents different difficulties for natural language processing, and Arabic is no exception. Arabic presents various unique problems that affect NLP work compared to languages like English; these three primary categories might help to explain these difficulties: linguistic, orthographic, and dialectal obstacles.

Arabic has a complicated language structure with a sophisticated morphological system based on roots and patterns, generating notable lexical variety. This complexity affects tokenization since Arabic words sometimes comprise several morphemes, such as clitics and inflectional endings, combined into a single token. The difficulty of Arabic arises from the necessity to precisely remove affixes while maintaining the fundamental meaning of the word, which calls for an awareness of the root-based morphology [2], [3].

Arabic's non-concatenative morphology makes lemmatization, the process of determining a word's base form, particularly difficult. Roots in Arabic are made of consonants mixed with vowel patterns to generate different meanings, usually in complex, non-linear patterns. Because roots and patterns do not always line up exactly, morphological study becomes difficult. Consequently, hybrid approaches combining machine learning with rule-based techniques have been developed to increase the accuracy of Arabic lemmatization and other NLP tasks [2], [3].

. Arabic's morphological diversity, marked by the great use of roots and patterns, generates significant vocabulary variance. The fact that words sometimes have several morphemes, such as clitics and inflectional endings, within a single token complicates tokenization. Stemming is also difficult since it calls for exact affixes to be removed while preserving the fundamental meaning of the word, therefore requiring a thorough knowledge of root-based morphology[4].

Dialectal diversity accentuates the complexity of Arabic NLP even more. Although Modern Standard Arabic (MSA) is the official standard in writing and media, many regional dialects—e.g., Egyptian, Levantine, and Gulf—are common in daily conversation. These dialects provide more challenges for NLP systems since their vocabulary, syntax, and pronunciation differ greatly. This phenomenon, called diglossia, presents a special difficulty since systems have to consider the significant linguistic difference between MSA and colloquial forms. Accurate language processing depends on dialect identification since geographical and socioeconomic variations among dialects hamper NLP tasks such as sentiment analysis, machine translation, and automatic speech recognition [5], [6], [7].

Developing thorough language resources for Arabic dialects still presents a challenge, mostly because of the limited availability of annotated corpora and linguistic instruments unique to every dialect. Researchers have been increasingly looking to social media sites like Twitter and Facebook to create dialect-specific corpora, which offer significant real-world language data for NLP model training [8], [9].

By allowing models to learn from data rather than depending just on handwritten rules, the area of machine learning, especially deep learning, has fundamentally changed NLP [9]. Text categorization, sentiment analysis, language modeling, and machine translation are just a few of the NLP chores where machine learning approaches have propelled progress. Deep learning models such as Transformers, Recurrent Neural Networks (RNNs), and Convolutional Neural Networks (CNNs) have set new benchmarks in NLP by attaining state-of-the-art performance on many tasks. Especially transformer models such as BERT (Bidirectional Encoder Representations from Transformers) and their successors (e.g., RoBERTa, XLM-R) use self-attention techniques to gather subtle contextual information inside the text, hence augmenting performance standards across NLP benchmarks [10].
In machine learning, class imbalance presents major difficulties, particularly in classification activities. When one class greatly outnumbers another, models often develop a bias toward the majority class, which results in suboptimal performance on the minority class, which can be especially important in applications where the minority class bears greater relevance. Class weighting helps to balance model attention and supports improved overall performance on imbalanced datasets by imposing larger penalties for misclassifying instances of the minority class [11].

Still, figuring out ideal class weights is difficult and usually calls for extensive research. Although this strategy ignores intra-class similarities that can affect learning, standard approaches sometimes assign class weights inversely proportional to class frequencies. While too diverse cases might contribute noise that obscures significant patterns, highly similar instances inside a class can cause overfitting since the model may acquire redundant features. Therefore, to attain strong model performance, a good method has to take intra-class similarity into account as well as class imbalance [10].

New ideas to solve these problems have been presented by recent studies. Promising in enhancing model outcomes are techniques such as staged learning, which incrementally balances the training data, and advanced models like ProtoBERT, which uses class centroids inside a learned BERT-based feature space. Furthermore, useful in improving minority class performance in imbalanced environments are specific loss functions such as label-distribution-aware margin loss (LDAM) [12].

These developments highlight how dynamically NLP is changing and the continuous efforts to overcome obstacles in training deep learning models on imbalanced data, therefore opening the path for more fair and accurate NLP solutions.

The key contributions of this paper are as follows:

1- We propose a novel technique to mitigate class imbalance by computing intra-class similarity using contextual text embeddings from BERT-based models and cosine similarity.
2- We incorporate both the intra-class similarity and the class instance distribution into a unified class-weighting strategy.
3- We introduce an adaptive class-weighted loss function to enhance the performance of BERT-based classifiers on imbalanced Arabic datasets.
4- We validate the effectiveness of our method through comprehensive experiments on two publicly available Arabic datasets: MSRPC and NSURL.

This work is arranged mostly as follows: Section 2 offers background data pertinent to the research; Section 3 examines relevant work in the literature, therefore orienting our method in the larger framework of past studies. Section 4 details the datasets we used for our experiments. Section 5 describes the approach of the suggested model; Section 6 follows with the results and provides a thorough analysis of the data. Section 7 ends the report and provides guidelines for further efforts at last.

II. BACKGROUND

Arabic NLP is a necessary subject in computer linguistics because of the language's particular grammatical complexity and general global usage. Although Arabic is among the most often used languages in the world, Arabic NLP has fallen behind English. Arabic's complex morphological structure, great dialectal variation, and strong inflectional patterns—all of which impair language processing—help to explain this discrepancy. Furthermore, embedding development in Arabic NLP has restricted access to premium resources, including annotated corpora and sophisticated computational tools [13], [14].

For basic NLP chores, including tokenization, stemming, lemmatization, and machine translation, Arabic's linguistic nuances—including varied dialects, rich morphology, and orthographic variations—pose significant obstacles. These difficulties call for specific tools and approaches that consider Arabic's unique characteristics. Dealing with these difficulties calls for both a strong awareness of Arabic linguistics and the creation of focused NLP solutions catered especially to Arabic's structural and dialectal characteristics [15].

Given these difficulties, it is becoming clear that Arabic NLP research requires more solid and all-encompassing materials if it is to progress. Bridge this gap and enable efficient processing across the several facets of the Arabic language by a strategic focus on creating specialized tools and enhanced language resources.

Based on its Semitic linguistic foundation, Arabic's morphology depends on a root-and-pattern system whereby words are produced by interweaving roots, usually made of three consonants, into particular patterns to generate different meanings. Extensive word diversity from a single root introduced by this non-linear morphological structure increases processing difficulty. To convey grammatical relations and meanings, prefixes, suffixes, and infixes are also included in roots, hence generating many forms for a single lexeme. Morphological analysis is essential in many applications since this complex morphological structure affects NLP operations like stemming and lemmatization [16].

Arabic's cursive script implies that each letter assumes several forms based on its place inside a word- initial, medial, final, or isolated—adding complexity to word segmentation and character recognition tasks. Arabic, which is written from right to left, offers additional difficulties for NLP systems originally intended for left-to-right languages. Furthermore, often left out in written language are short vowels or diacritics, which NLP systems have to negotiate [17], causing uncertainty in word meaning and pronunciation.

Arabic includes, in addition to Modern Standard Arabic (MSA) for formal settings, a range of dialects, including Egyptian, Levantine, and Gulf dialects, which are used in daily communication. This diversity produces notable lexical, syntactic, and phonetic heterogeneity that complicates the development of NLP tools that run consistently across languages. Strong Arabic NLP [18] depends on comprehensive models that reflect the variances in dialects since they differ greatly.

Dealing with these difficulties calls for a thorough knowledge of Arabic's linguistic features as well as the creation of specific computational instruments. Through both knowledge-based and empirical approaches, advances in Arabic computational morphology have helped to advance NLP; nonetheless, important difficulties still exist [16], [17].

Dialectal NLP and Dynamic Data Requirements: Large annotated corpora and advanced models capturing dialectal subtleties are prerequisites for NLP systems able to process Arabic dialects. Dialectal Arabic, on the other hand, is not standardized and develops rapidly; hence, regular model adjustments are needed. Dialectal Arabic is a dynamic field of research within NLP due in part to this lack of standardization [18].

The introduction of machine learning, especially deep learning, has revolutionized NLP by moving from rule-based systems depending on extensive human feature engineering to data-driven approaches directly extracting patterns from data. While deep learning, especially with neural networks, has surpassed conventional techniques in many NLP applications, machine learning enables automated model training using task-specific data [19], [20].

Early machine learning techniques included logistic regression, support vector machines (SVMs), and decision trees in NLP applications that needed significant feature engineering from domain specialists. Common features for text classification projects were syntactic dependencies, part-of-speech tags, n-grams, and TF-IDF. Although somewhat successful, these hand-crafted elements sometimes fail to grasp the deep semantics and contextual complexities of language, therefore restricting performance on challenging tasks, including sentiment analysis and machine translation [21], [22].

Machine learning and deep learning have transformed many disciplines, including computer vision and numerical data classification, while convolutional neural networks (CNNs) have revolutionized computer vision across many domains by enabling sophisticated image recognition, object detection, and image segmentation [23], [24], [25], [26], [27]. Fully connected neural networks have advanced numerical data classification tasks [28].

Deep learning architectures, especially neural networks, have changed feature extraction in NLP by allowing models to learn hierarchical representations straight from raw input. Early deep learning uses in natural language processing made use of recurrent and convolutional neural networks. Originally intended for image processing, CNNs suited well to text classification by considering text as a one-dimensional string of words or characters, capturing local patterns and relevant structures through convolutional filters [8], [29], [30], [31], [32]. Developed for sequential data, recurrent neural networks (RNNs), especially Long Short-Term Memory (LSTM) networks, showcased their efficacy in applications needing contextual awareness, including machine translation and language modeling. Crucially for capturing meaning in longer text, LSTMs also solve the vanishing gradient problem by allowing the model to preserve long-range dependencies[33].

With architectures like BERT (Bidirectional Encoder Representations from Transformers), Transformer models represented a major leap forward in NLP and set new benchmarks. Transformers outperform RNNs in modeling long-term dependencies by using self-attention processes to seize linkages between words at different distances within a sentence. Originally trained on big text corpora and refined for certain tasks, BERT and its variants (e.g., RoBERTa, XLM-R) have produced state-of-the-art results in applications including question answering, text summarizing, and machine translation, so considerably improving NLP capabilities [34].

In machine learning, class imbalances are a recurring problem, especially in classification problems where one class is much underrepresented. The main issue in applications like fraud detection, medical diagnosis, and anomaly detection, with imbalanced datasets, is the bias of models towards the majority class, therefore compromising the performance of the minority class[35].

Class weights can be given in the loss function during training to offset this by applying a higher penalty for misclassifications of the minority class, and so encouraging balanced learning. Usually, class weights are adjusted inversely proportional to class frequencies or use Synthetic Minority Oversampling Technique (SMOTE) to balance the dataset prior to training. These weights are crucial in binary classification problems when one class is much more common to prevent the model from defaulting to majority predictions, thereby producing high accuracy but poor minority class performance [35], [36].

Standard class-weighting techniques, however, can ignore data-specific properties—that is, the degree of similarity between cases within a single class. The high intra-class similarity could cause overfitting since the model might specialize too much on repeating features. On the other hand, poor intra-class similarity can generate noise, so the model finds it more difficult to learn relevant patterns [10], [37].

Our proposed method for class weight computation is designed to address both traditional class imbalance and intra-class semantic similarity. Unlike conventional approaches that rely solely on the frequency of class instances, our method incorporates the internal structure of each class by evaluating how semantically similar the samples are within that class. The process begins by dividing the dataset into separate subsets based on class labels. For each class, we compute the total pairwise cosine similarity between the contextual embeddings of its instances, generated using a pretrained transformer model (e.g., XLM-Roberta-Large).

The rationale is that a class with highly similar samples contains more redundancy and contributes less unique information to the learning process. Conversely, a class with lower intra-class similarity contains more diverse information and is potentially more valuable for generalization. Based on this insight, we assign class weights in proportion to the inverse of the aggregated intra-class similarity. Specifically, the class with the lower total similarity is given a normalized weight of 1, while the other class is assigned a smaller weight proportional to the similarity ratio between the two.

This adaptive weighting scheme ensures that the model does not disproportionately focus on redundant information. Instead, it emphasizes the learning of distinctive and varied features, especially from classes with greater internal variability. By integrating intra-class semantic structure with class frequency considerations, our method enhances model robustness and improves classification performance in scenarios where semantic overlap varies significantly between classes.

Developing robust Arabic NLP tools remains a significant challenge due to the language's complex morphological structure, diverse dialects, rich vocabulary, and orthographic variations. Although machine learning and deep learning have led to considerable progress in Arabic text classification tasks, achieving reliable performance, particularly in imbalanced datasets, requires more than just large models and data. Traditional class-weighting strategies typically address imbalance by assigning weights inversely proportional to the number of instances per class. However, such methods ignore an important factor: the semantic similarity among samples within each class, which can significantly influence the learning dynamics of neural models.

To address this, we propose a novel class-weighting method that not only considers class imbalance in terms of sample counts but also captures the intra-class semantic similarity. Specifically, we embed each sentence using a transformer-based language model (XLM-Roberta-Large), and then

compute pairwise cosine similarity scores between all samples within each class. These similarity scores provide an estimate of the internal redundancy of the class—i.e., whether the class is composed of semantically repetitive or diverse examples. We aggregate these pairwise similarities to compute a total intra-class similarity score for each class.

Our proposed weighting mechanism then combines these similarity-based scores with class frequency to derive adaptive class weights. The class with lower total similarity (i.e., more semantic variation) is assigned a higher weight (normalized to 1), while the class with higher similarity is assigned a proportionally reduced weight. This approach allows the model to pay more attention to classes that offer greater information diversity during training, leading to improved representation learning and generalization.

This enhanced weighting strategy is particularly useful in the context of Arabic datasets, where semantic richness and label ambiguity are common. We validate the effectiveness of this method using two binary Arabic datasets—the Microsoft Research Paragraph Corpus (MSRPC) and Semantic Question Similarity in Arabic (NSURL)—both of which present unique challenges in terms of class balance and linguistic diversity. The results confirm that our technique not only improves classification performance but also promotes fairer learning across semantically diverse classes.

III. RELATED WORK

Class imbalance has been a longstanding challenge in text classification, and numerous approaches have been proposed to address it in both traditional and deep learning settings. The reviewed works below represent key strategies, including oversampling, cost-sensitive learning, and the use of pre-trained language models. While they demonstrate effectiveness in general contexts, most overlook the structural and semantic characteristics of low-resource or morphologically rich languages like Arabic.

Krawczyk et al. [38] proposed Radial-Based Oversampling (RBO), which enhances multiclass classification by generating synthetic instances arranged radially around class centroids. The method effectively improves minority class representation in numerical datasets by preserving class boundaries. However, RBO was designed for structured feature spaces and does not leverage semantic or linguistic features, making it less suitable for high-dimensional language embeddings or textual contexts.

Wang and Yao [39] offered a comprehensive treatment of multiclass imbalance problems, categorizing typical issues and proposing strategies such as resampling and cost-sensitive classification. Their framework helped shape foundational thinking in the field. However, the solutions they suggest are largely statistical and do not account for the internal semantic composition of each class, which can have a strong influence on learning outcomes in text classification tasks.

Taha et al. [40] introduced a combined over-sampling and under-sampling framework for multilabel text classification, incorporating class alignment to handle label co-occurrence imbalance. Their approach is tailored for text and effective in multilabel scenarios, but it does not use embedding-based representations or capture intra-class semantic variation. Additionally, their experiments were not extended to complex low-resource languages.

Rafi-Ur-Rashid et al. [41] tackled the issue of class imbalance in Bangla text classification using data augmentation and BERT fine-tuning. Their solution reflects an awareness of under-resourced language challenges and leverages powerful pre-trained models. Nevertheless, the reliance on data augmentation can introduce noise and requires task-specific heuristics. Furthermore, it does not consider redundancy or diversity within classes during training.

Shaikh et al. [42] addressed imbalance in highly skewed English text datasets by employing deep neural language models, including BERT. Their work showed that transformer-based models benefit from traditional weighting strategies, such as using inverse class frequencies. However, the method does not investigate the internal structure of classes or adapt weights based on semantic redundancy among class samples.

In contrast to the above studies, our method introduces a class-weighting strategy that explicitly considers intra-class semantic similarity. Using contextual embeddings from a transformer model, we compute cosine similarity among samples within each class and adjust weights accordingly. This allows the model to focus more on semantically diverse and informative examples during training. While prior studies largely overlook this factor, which is important for morphologically rich languages.

Although class imbalance techniques are well developed in general NLP, most prior work was not designed for Arabic text. The Arabic language presents unique challenges, including root-based morphology, high lexical variation, and dialectal diversity, which often lead to semantically redundant instances within classes. Our proposed method directly addresses these challenges by incorporating intra-class semantic variation into class weight computation, offering improved performance on Arabic datasets without relying on augmentation or synthetic sampling.

In the following subsections, we review related works that have been applied to the two benchmark datasets used in this study, namely MSRPC and NSURL, where we can compare

our proposal with them. The MSRPC and NSURL datasets will be presented and discussed in detail in the next section.

In the next part of this section and as explained in Figure 1, we present summary of the previous work related to semantic similarity approaches on the two datasets MSRPC and NSURL.

A. DATASET 1: MICROSOFT RESEARCH PARAPHRASE CORPUS (MSRPC)

1) Similarity-Based Methods

Corley and Mihalcea's approach is a prominent method for assessing the semantic similarity between two text sections by categorizing the words into distinct groups—nouns, verbs, adjectives, and adverbs. Following this, each word in one text is matched with related words in the other text, enabling a focused examination of their commonalities. This approach has an accuracy level of 71.5% [43].

Another work suggests a dual-strategies approach to evaluate semantic similarity between texts by using both knowledge-based and corpus-based similarity measures. Previous studies mostly concentrated on either huge texts (for text categorization and information retrieval) or individual words (by means of synonym tests). This approach, on the other hand, highlights the semantic similarity of short texts since it acknowledges that, both online and offline, a good amount of the material that is accessible consists of little morsels like abstracts of scientific papers, picture captions, and product descriptions. Under this method, the accuracy came out to be 70.3% [44].

Under another technique, a textual entailment detection system was strengthened using a graph-based methodology. Here, both text sections are converted into dependency networks so that the researchers may find similarities depending on common edges and vertices. This approach finds whether one text passage suggests the other; if both segments suggest one another, they are considered paraphrases. With an accuracy of 70.61% [45], tests run using the MSRPC corpus produced findings on par with those obtained with other similarity-based methods.

Furthermore, a technique using weighted matrix factorization (WMF) was suggested to represent text fragments. This method used low weights for missing words to replicate their existence rather successfully. With an accuracy score of 71.51% [46], the results of this method for the MSRPC corpus matched those of past similarity-based techniques.

Another method assesses the semantic relatedness between two texts using the Synset shortest path metric from WordNet. The authors classified the two segments as paraphrases and assessed sentence-level semantic relatedness by determining the shortest path between words in each segment. The two sections were considered paraphrases if the relatedness score passed a set criterion. This approach mostly emphasizes word-level analysis and attained a 71.1% [47] accuracy.

2) Classification-Based Methods

A suggested method evaluates the semantic similarity between phrases using similarity measurements as features. The two sentences are first transformed into their canonical versions. After that, different lexical commonalities among the surface books are calculated. At last, a decision tree distinguishes the statements as paraphrases or not. Nevertheless, even using a supervised learning strategy, the outcomes of this method were rather poorer than those of other machine learning approaches. This can be ascribed to the loss of instructive components during the conversion to a canonicalized form, therefore producing an accuracy of 71.9%[48].

Still, another creative method combines word similarity measures with word overlap. This approach uses three distinct classifiers: maximum entropy, K-nearest neighbor (KNN), and support vector machine (SVM). Using the MSRP corpus, the researchers investigated many feature combinations, including word overlap, word similarity, and hybrid models using both measures. Combining the outputs of the classifiers using a voting system produced the best results—an accuracy of 76.64% [49].

Another work concentrated on an SVM classifier using several feature sets, including dependent features, BLEU scores, lemmatized BLEU features, n-gram overlap, and extra BLEU features. The best feature combination omitted just the lemmatized n-gram. The dependence characteristics provided high-precision findings, underlining their importance in spotting bogus paraphrases, and obtained an accuracy of 75% [50].

Another study suggested a two-step approach for paraphrasing identification. The first stage uses Semantic Role Labeling (SRL) and parsing to gather predicative argument tuples. Then, using a greedy approach, pairs of similar tuples are found from the two phrases. In the second stage of evaluation, any unpaired tuples are sent to an SVM classifier. A sentence is said to be paraphrased if it has a few important unpaired tuples and a large number of paired tuples, therefore attaining an accuracy of 72.0% [51].

Another method applied a maximum entropy classifier based on three separate sets of similarity features. The first collection consisted of 133 features obtained from applying nine similarity measures to 10 distinct abstractions of the original texts. The second set viewed shared terms as identical; the third set merged the second set with three extra dependency traits. The third feature set testing on the MSRP corpus revealed the top performance. Two different feature selection techniques were used, thereby reducing features that very slightly affected the results and obtaining an accuracy of 76.17% [52].

To find paraphrases, a new quasi-synchronous dependency grammar was suggested, combining lexical semantic features

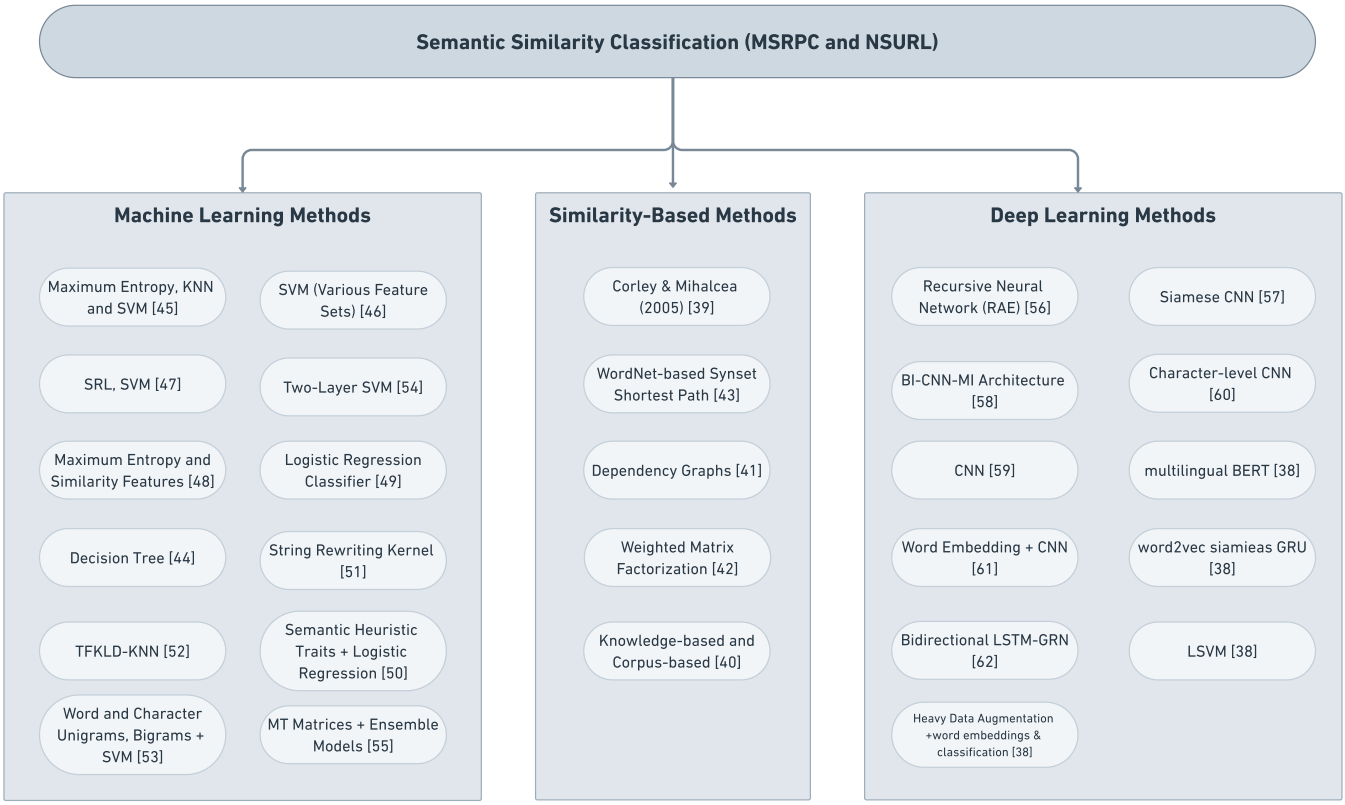


FIGURE 1: Taxonomy for Semantic Similarity Approaches on MSRPC and NSURL

with syntax. This model obtained an amazing accuracy of 80.41% [53] when linked with a logistic regression classifier.

One also used semantic heuristic features for paraphrasing identification. Using a logistic regression classifier and many feature vectors tested on the MSRP corpus [54], this method categorized sentence pairs as paraphrases.

Another original method for paraphrasing was using an SVM classifier. This work presented a novel string rewriting kernel measuring the degree of similarity between two sentences. It generated findings compatible with current machine learning methods and obtained an accuracy of 76.3%[55].

An improved SVM method was developed, integrating continuous and discontinuous phrases into word embeddings. Furthermore, suggested by the researchers is TFKLD-KNN, a variation on TF-KLD in which word and phrase vectors are weighted depending on their separating power between the two sentences. The sentences are categorized with SVM and shown as weighted vectors. The results showed how effectively phrase embeddings might be included to produce an accuracy of 78.7% [56].

Using word and character unigrams and bigrams as features in an SVM classifier provides still another approach for paraphrasing identification. The RBF kernel proved better than the linear counterpart when the researchers tested radial basis function (RBF) and linear kernels. With five distinct corpora—including the MSRP, Twitter Paragraph, PAN, Turkish Paragraph, and Russian Paragraph—they tested their technique, obtaining an accuracy of 74.2% on the MSRP corpus [57].

Another suggested method for paraphrasing is a two-layer SVM classifier. While a single classifier in the higher layer makes the ultimate decision, the lower layer consists of multiple classifiers, each trained to identify a particular form of paraphrasing. Using MT measurements, dependency overlap, and other n-gram overlap characteristics, this method obtained an accuracy of 77.86% [58].

Another approach fed into an ensemble of three classifiers—logistic regression, SVM, and instance-based classifiers—eight separate machine translation (MT) matrices as features for paraphrasing accuracy of 77.4% [59] came from testing on the MSRP corpus.

Paraphrase identification included a deep learning method for feature representation. To derive features from an unlabeled parse tree, the authors suggested either an unfolding recursive autoencoder (RAE) or a recursive neural network. By means of vector reconstruction of words and sentences, the RAE enables the computation of a similarity matrix between the two texts. Testing on the MSRP corpus, a SoftMax

classifier then finds whether the phrases are paraphrased, therefore attaining an accuracy of 76.8%[60].

Two different architectures were proposed: the first one uses a single Siamese convolutional neural network (CNN) for sentence feature representation. Then, an MLP evaluates the degree of output matching. This architecture, however, delays interaction between the phrases until after their representations are finished, therefore perhaps losing vital information. The authors created a second architecture with a single CNN that simultaneously processes a mixture of the two texts concurrently, feeding the resulting feature representations into an MLP for matching assessment, therefore attaining an accuracy of 69.9% [61].

A BI-CNN-MI architecture was proposed for paraphrasing vernacular identification: MI denotes multi-granularity, and BI-CNN refers to a pair of CNNs. Along with a third CNN that combines these representations at several levels, this architecture includes two Siamese CNNs, creating representations for the studied sentences. Applied to the MSRP corpus with an accuracy of 78.1% [62], the final integrated characteristics feed into a logistic regression classifier and produce somewhat interesting results.

Another method used CNN to extract features from books at several granularities. Using similarity metrics, the authors compared the learned representations of the two sentences using several pooling methods and convolutional processes. With an accuracy of 78.60% [63], our approach yielded findings in line with past deep-learning approaches.

A new method proposed a CNN-based sentence semantics representation using character-level input. Two different character encodings are fed into separate CNNs to generate representations for every sentence, which are subsequently concatenated into a single vector instead of conventional word embeddings. After that, the sentences are categorized as paraphrases depending on this last vector using a logistic regression classifier. This method may be used in any language and does not depend on knowledge of syntactic structures or word meanings; yet, it did not perform as effectively as prior CNN approaches using word embeddings, obtaining an accuracy of 73.3% [64].

Another study used CNNs to find feature vectors by considering both similarities and differences in sentence pairs. First, one models the two sentences with word embedding vectors. To produce a new vector reflecting the sentence pair, each word vector from one sentence semantically matches each word vector from the other sentence. Following a CNN, this vector is then separated into dissimilarity and similarity vectors, which are merged into a single feature vector. By use of a similarity score computed from this feature vector, the ultimate decision on whether the two sentences are paraphrased rests on an accuracy of 78.4% [65].

Using Sklearn's Vectorizer Analyzer across three configurations—word-level, character-level, and character-with-word-level—the Speech Translation Team compiled a comprehensive feature set. For every set, they investigated the applications of n-grams (1, 2, 3, 4, and 5). Complementing word stemming and POS tagging, they used punctuation removal, stop word filtering, and text-normalizing pre-processing tools. The researchers assessed several classifiers, including BNB, LogReg, LSVM, MNB, PassAgg, PRP, SGD, and CNN, finally obtaining the best results with the LSVM classifier [66].

In [67], the authors presented a gated relevance network (GRN) integrated bidirectional long short-term memory (Bi-LSTM) model. This method captures spatial representations of text fragments by means of a recurrent neural network. While a pooling layer chooses the most instructive interactions, a gated relevance network finds the semantic links between several sections. Then, classification is done by a multilayer perceptron (MLP), with an accuracy of 80.92%. This method was among the most successful ones tested on the MSRP corpus.

B. DATASET 2: SEMANTIC QUESTION SIMILARITY IN ARABIC, SOLUTIONS FOR UNDER-RESOURCED LANGUAGES (NSURL)

Using a range of deep learning methods, mostly drawing on the BERT model, this team created. They concentrated on improving the multilingual BERT model by means of sentence similarity task performance. They aimed to find the ideal parameter set that produced the best predictions by varying many hyperparameter combinations and iterating the tests using different random seeds. Eventually, by averaging the prediction results from several different studies, they developed a single model. Three models from their collection outperformed on the public dataset; on the private dataset, configurations of four, five, and six models showed better performance [66].

This team effectively expanded the data size to 45,514 samples by using a strict pre-processing technique involving punctuation removal and dataset augmentation based on established rules. For input, they extracted contextual word embeddings from the pre-trained Arabic Elmo model. Their architecture consisted of a merging layer and a sequence representation extractor, followed by a four-layer deep neural network dedicated to classification [66].

Onekaggler Team: Specifically employing the aforementioned tweets_cbow_300 embeddings, this method created a powerful deep learning model with two input layers—one for each question—paired with a shared, trainable word embedding layer generated from the word2vec model. Three stacked bidirectional GRU layers with 256, 128, and 64, respectively—hidden node configurations—were part of the architecture. By use of a dot product of the outputs from the last levels of both questions, the output layer computed the

cosine similarity. In concert with the Nesterov Adam optimizer [66], they used mean squared error as the loss function.

Figure 1 shows a thorough taxonomy of the several methods applied for the assignments. This number captures the methodologies used, stressing the primary tactics and tools applied in several studies, and thus offers a clear and orderly summary of the several approaches in this field of research.

Despite the effectiveness of various methods for handling class imbalance, most existing approaches rely heavily on adjusting class weights or resampling techniques based solely on class frequency. However, these methods often overlook an important factor in text classification: the semantic similarity or redundancy within each class. In many cases, a class may contain a large number of samples, but if those samples are semantically similar, they contribute less informative diversity to the learning process. On the other hand, a smaller class with more semantically diverse samples may be more valuable for learning robust representations. This observation highlights a key limitation in prior work. To address this, we propose a novel class-weighting approach that not only accounts for the number of samples per class but also incorporates intra-class semantic similarity into the weighting scheme. By combining these two dimensions, our method aims to achieve a more balanced and semantically aware training process that improves classification performance, particularly in the context of complex and morphologically rich languages such as Arabic.

IV. DATASETS

We chose the two datasets, MSRPC and NSURL, according to their fit for our particular work requirements. Both datasets seek to assess semantic similarity, thereby establishing if two sentences have the same meaning and are set for binary classification. These datasets are quite appropriate for our goals since they solve similar language processing difficulties in the Arabic language environment by means of their concentration on semantic equivalency in pairs of Arabic phrases.

A. DATASET 1: MICROSOFT RESEARCH PARAPHRASE CORPUS (MSRPC)

In the subject of paraphrase vernacular detection, the Microsoft Research Paragraph Corpus (MSRPC) [68] is a generally accepted benchmark collection. It comprises pairs of sentences tagged with binary classifications denoting whether they are paraphrases of one another or transmit the same semantic meaning. Within the NLP community, this dataset has been thoroughly examined, and several approaches aiming at improving the accuracy of paraphrase erosion have emerged.

With 1,331 sentence pairs overall in its Arabic translation, the MSRPC consists of 703 not paraphrased and 628 records classified as paraphrased. We split the dataset into a training set and a testing set with a distribution of 70% for training and 30% for testing to help facilitate evaluation. The training collection comprises specifically 494 non-paraphrased and 437 paraphrased pairs, for 931 records. With 201 paraphrased pairs and 209 non-paraphrased pairs included in the test set, 400 records result.

Table 1 shows the overall dataset distribution; Figure 2 shows the training subset distribution. Figure 3 shows the distribution inside the subset for testing visually. Figure 4 also shows instances from the dataset, including both Arabic statements and their English translations, therefore illustrating the nature of the paraphrasing.

TABLE 1: DATASET 1 DISTRIBUTION

Set	Paraphrased	Not Paraphrased	Total
Train	437	494	931
Test	191	209	400
Total	628	703	1,331

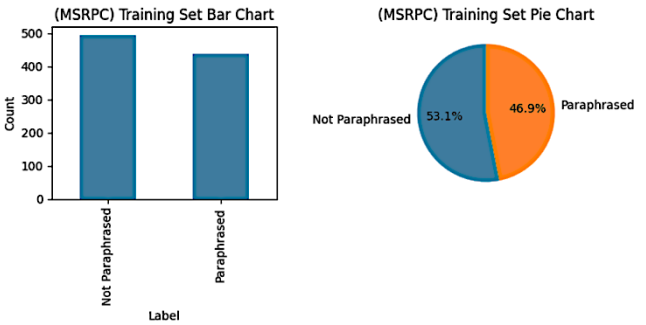


FIGURE 2: Dataset 1 Training Distribution

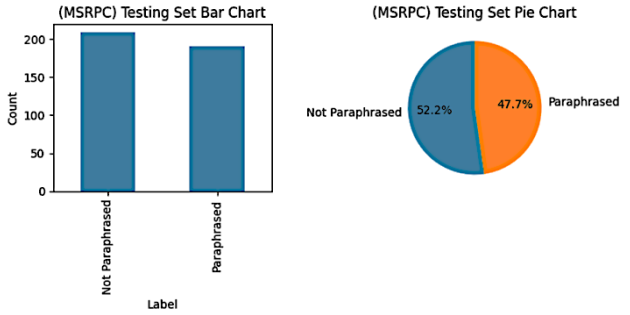


FIGURE 3: Dataset 1 Testing Distribution

Label	Sentence 1	Sentence 2
Paraphrased Paraphrased (translated)	قالت أن دروس مازال محتجزا في السجن، وهو الآن في غرفة العزل She said that Dross is still detained in prison, and is now in an isolation room	تم وضع دروس الليلة الماضية في غرفة العزل في السجن نفسه Last night, Dross was held in the isolation room of the prison itself
Not Paraphrased Not Paraphrased (translated)	أدخلت شركة فيريساين خدمتها لموقع المستكشف في 15 سبتمبر VeriSign introduced its Explorer Site service on September 15	تستعر المعركة حول فيريساين على خدمة الموقع المكتشف منذ ثلاثة أسابيع The battle over VeriSign over the Discovered Location service has been raging for three weeks
Paraphrased Paraphrased (translated)	أعلنت شركة النخيل الأربعاء عن خطط للحصول على هاندسبرينغ وهي شركة شقت طريقها عن طريق جيف هوكينز، الذي يعتبره الكثيرون الأب لشركة بالم On Wednesday Nakheel Company announced its plans to acquire Handspring, a company created by Jeff Hawkins, considered by many to be the father of Palm	أعلنت شركة النخيل يوم الأربعاء أنها تعتزم شراء شركة هاندسبرينغ ، الشركة التي تم إنشاؤها من قبل المنشق بالم المؤسس المشارك جيف هوكينز Palm announced on Wednesday that it intends to buy Handspring, the company created by renegade Palm co-founder Jeff Hawkins
Not Paraphrased Not Paraphrased (translated)	نوهت الصحيفة أن مستثمري جوجل هي مؤسسات رأسمالية بارزة منها : كلينر بيركنز كوفيلد ، بايرز وسيكويا كابيتال The newspaper noted that Google's investors are prominent capitalist institutions, including: Kleiner Perkins Caufield, Bayers, and Sequoia Capital	داعمو جوجل في مراحلها الأولى هم :جامعة ستانفورد ومقرها كاليفورنيا، الشركات الرأسمالية كلينر بيركنز وسيكويا كابيتال Google's supporters in its early stages are: California-based Stanford University, venture capital firms Kleiner Perkins and Sequoia Capital

FIGURE 4: Dataset 1 Example with Arabic and translated English Examples

TABLE 2: DATASET 2 DISTRIBUTION

Set	Similar	Not Similar	Total
Train	5,397	6,600	11,997
Test	1,718	1,997	3,715
Total	7,115	8,597	15,712

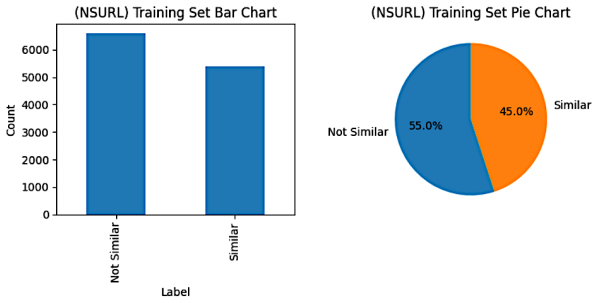


FIGURE 5: Dataset 2 Training Distribution

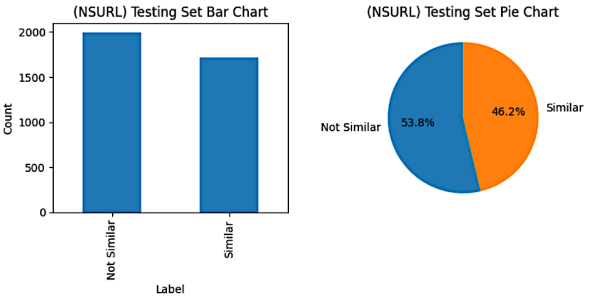


FIGURE 6: Dataset 2 Testing Distribution

B. DATASET 2: SEMANTIC QUESTION SIMILARITY IN ARABIC, SOLUTIONS FOR UNDER-RESOURCED LANGUAGES (NSURL)

Two main uses in natural language processing (NLP) that greatly help to solve the difficult problem of question semantic similarity, sometimes referred to as Q2Q, are question-answering systems and duplicate question identification. This work was developed for the first NLP Solutions for Under-Resourced Languages workshop (NSURL 2019) [66]. Its main goal is to determine if two differently phrased queries semantically match.

There are 15,712 Arabic question pairings in total for this study; 7,115 records are rated as similar and 8,597 records as not comparable. Training and testing subgroups methodically separate these pairs. The training set comprises 5,397 similar pairings and 6,600 not comparable pairs, resulting in a total of 11,997 records. The test set comprises 1,718 similar pairings and 1,997 non-similar pairs, resulting in 3,715 records.

Table 2 describes the whole dataset; Figure 5 shows the distribution of the training subset. Figure 6 graphically shows the distribution within the test subset. Moreover, Figure 7 presents instances from the dataset, including Arabic sentences and their English translations, illuminating the nature of the Q2Q task [66] and its applicability for further Arabic NLP research.

Label	Question 1	Question 2
Similar	ما مناخ مانتيتوبا السائد؟	ما هو المناخ السائد في مانيتوبا؟
Similar (translated)	What is Manitoba's climate?	What is the climate in Manitoba?
Not Similar	كيف احضر الشكشوكة على الطّريقة التونسيّة ؟	من طرق تحضير الشكشوكة ؟
Not Similar (translated)	How do I prepare shakshuka the Tunisian way?	How to prepare shakshuka?
Similar	ماذا نعني بوسائل الاتصال الحديثة؟	ما هي وسائل الاتصالات الحديثة؟
Similar (translated)	What do we mean by modern means of communication?	What are modern means of communication?
Not Similar	ما طريقة تحضير التمر بالطحينية ؟	من طرق تحضير كيكة التمر؟
Not Similar (translated)	How to prepare dates with tahini?	How to prepare date cake?

FIGURE 7: Dataset 2 Example with Arabic and translated English Examples

This work prepares the way for a more thorough exploration of the optimization of class weights and the construction of more sophisticated models able to negotiate the complexity of the Arabic language. Future research could expand our approach to other languages displaying comparable morphological and dialectical intricacies, therefore helping to develop natural language processing (NLP) more generally. By changing our methodology, academics could find fresh ideas and techniques that improve NLP capacities in various linguistic settings, thus promoting more efficient and sophisticated knowledge in multilingual environments.

V. METHODOLOGY

A. Problem Motivation

As was already mentioned, one major obstacle related to class weighting is that data imbalance can go beyond just the unequal distribution of events within classes. Depending just on the ratios and frequencies of events in every class could prove inadequate, since this method usually ignores the subtleties of intra-class similarities. Many people believe that a dataset is said to be balanced if its instances across classes are equal. Though this presumption seems reasonable, it ignores the complexity of intra-class similarity, which can significantly affect generalization and model performance. To show this problem, let us create a hypothetical dataset of four pairs of statements, designated S1 and S2. In this situation, the semantic similarities between the sentences in every pair can vary greatly, even if every class has an equal number of examples, which can cause possible misunderstanding and classification errors.

- Pair 1:
  - S1: What is the weather like today?
  - S2: What is the weather like on this day? [Similar]
- Pair 2:
  - S1: What is the weather like tonight?
  - S2: What will the weather be this evening? [Similar]
- Pair 3:
  - S1: How to make pasta?
  - S2: Will cars fly in the future? [Not Similar]
- Pair 4:
  - 1S: Will humans fly planes in the future?
  - S2: Will cars fly in the future? [Not Similar]

Couples 1 and 2 clearly show a great degree of resemblance. When a model is taught on these pairs, the weight changes will probably be modest and reflect their similarity. Pairs 3 and 4 offer a distinct picture: in pair 3, the two sentences lack similarity, whereas in pair 4, the sentences are comparable but do not transmit the same semantic meaning. As such, the degree of weight changes in pairings 3 and 4 is probably different, which emphasizes the need for intra-class differences in the training process. Our drive and basic presumption for doing this research come from this observation. We thus want to investigate these subtleties in our testing with the two previously mentioned binary classification datasets: the Semantic Question Similarity in Arabic (Solutions for Under-Resourced Languages, NSURL) and the Microsoft Research Paragraph Corpus (MSRpc). Investigating these datasets will help us to better grasp how intra-class similarity affects model performance and training.

B. Experimental Setup and Hyperparameters

In this study, the embedding model we employed was XLM-RoBERTa-Large, a transformer-based multilingual language model built upon the RoBERTa architecture and trained on a large-scale corpus comprising 2.5 TB of filtered CommonCrawl data covering 100 languages, including extensive Arabic content.

The XLM-RoBERTa-Large model was selected for three main reasons: (1) its ability to capture rich contextual and semantic relationships in morphologically complex languages such as Arabic, (2) its proven robustness in cross-lingual transfer scenarios, and (3) its strong performance on semantic similarity and classification benchmarks. To adapt XLM-RoBERTa-Large to our tasks, we performed full end-to-end fine-tuning rather than freezing the transformer layers. Fine-tuning was carried out using the AdamW optimizer with a learning rate of 1×10^{-5}, a dropout probability of 0.5 applied to fully connected layers, a batch

size of 64, and training for 40 epochs. The MSRPC dataset was used as the primary tuning dataset to identify optimal hyperparameters, which were then applied without modification to the NSURL dataset to maintain consistency and allow for fair cross-dataset evaluation.

The contextual embeddings were taken from the model's final hidden layer, representing each input sentence as a dense semantic vector. These embeddings served two critical purposes: (1) as input to the classification head during the fine-tuning stage, enabling the model to directly learn the classification boundaries, and (2) as the basis for computing pairwise cosine similarity between all samples within the same class in order to derive our proposed adaptive class weights. Using embeddings from XLM-RoBERTa-Large allowed us to integrate both syntactic and semantic features into the intra-class similarity computation, ensuring that the weighting scheme was informed by rich contextual representations rather than sparse or surface-level features. This combination of full model fine-tuning, consistent hyperparameter settings, and embedding-driven similarity computation was instrumental in achieving the performance gains reported in our results.

C. Text Embedding and Intra-Class Similarity Calculation

Multiple steps were taken in our method to the MSRPC dataset to accommodate intra-class similarity, therefore resolving the imbalance sometimes disregarded in traditional methods. The following pipeline presents our approach:

We used XLM- Roberta-Large, a robust transformer model well-known for its NLP task performance and excellent text representation capacity. This pre-trained model helped to embed the text data so that it might be converted into numerical form, fit for downstream analysis.

Given two classes—0 and 1—we split records for each class into separate arrays using intra-class cosine similarity calculation. We computed cosine similarity ratings for every record in every array against all other records in the same class and compiled these similarity values. This generated two aggregated cosine similarity values, one for each class, thereby reflecting intra-class similarity.

D. Adaptive Class Weight Computation

We computed modified class weights using these intra-class similarity scores, considering the aggregated similarity scores as approximators for class instance counts. This method lets us balance the internal similarity distribution inside every class as well as the frequency of classes, so it may improve model generalization and performance.

E. Model Training and Weight Integration

Following the determination of the ideal class weights, we fine-tuned the XLM-Roberta-Large model using the computed weights to improve the model's capacity to correctly differentiate between paraphrased and non-paraphrased samples.

F. Class Imbalance: Mathematical Formulation

Let a classification dataset be defined as:

$$D = \{(x_i, y_i)\}_{\{i=1\}}^{N}$$

Where:

- x_i represents an input sample,
- $y_i \in \{0, 1, \dots, C-1\}$ denotes its class label among C total classes.

Let N_c be the number of samples in class c, such that:

$$N = \sum_{c=0}^{C-1} N_c$$

The class distribution is said to be imbalanced when:

$$\exists\, c, d \in \{0, \dots, C-1\}, c \neq d \; such\; that \; \frac{N_c}{N_d} \gg 1 \; or \; \frac{N_c}{N_d} \ll 1$$

A commonly used Imbalance Ratio (IR) is:

$$IR = \frac{\max_c(N_c)}{\min_c(N_c)}$$

In a binary classification problem ($i.e., C = 2$), with class $y \in \{a, b\}$, let:

- N_0 is the number of samples in the majority class,
- N_1 is the number in the minority class,

Then:

$$IR = \frac{N_{majority}}{N_{minority}} = \frac{max\,(N_0, N_1)}{min\,(N_0, N_1)}$$

The imbalance can bias a classifier $f_\theta(x)$ toward the majority class by minimizing the standard cross-entropy loss:

$$L_{CE} = -\sum_{i=1}^{N} log\; P_\theta(y_i \mid x_i)$$

As a result, the model tends to focus on minimizing errors for the majority class, leading to poor performance on the minority class.

To compensate, class weights w_c are often introduced in the loss function:

$$L_{weighted} = -\sum_{i=1}^{N} w_{yi} . \log P_\theta(y_i \mid x_i)$$

Where $w_c = \frac{1}{N_c}$ or a normalized variant.

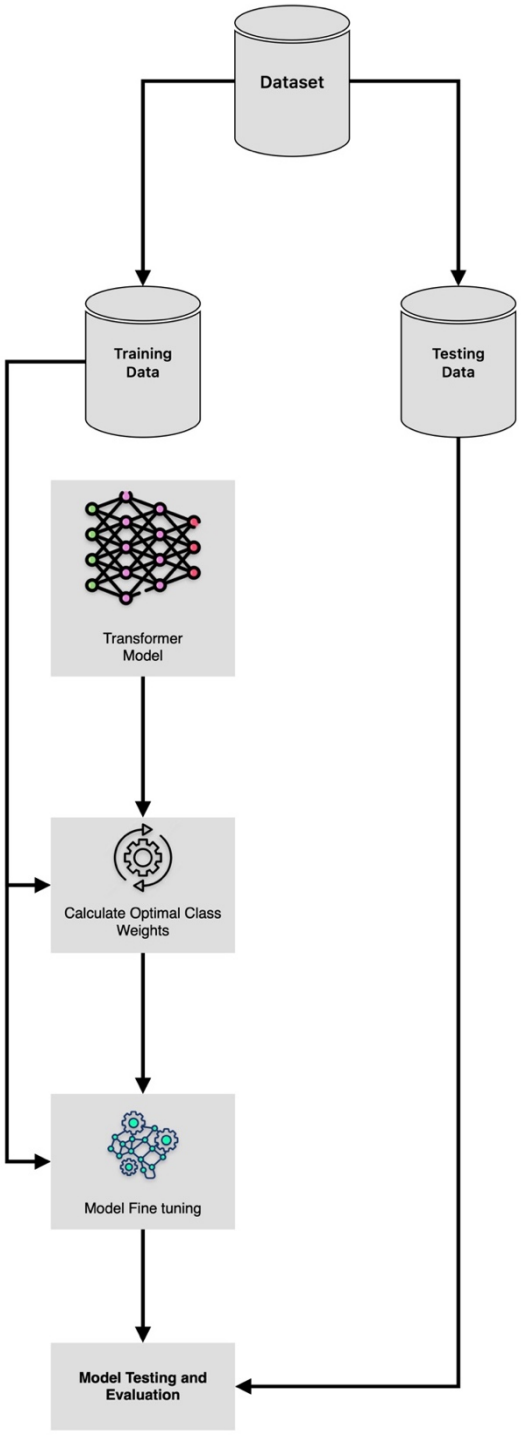


FIGURE 8: Approach Pipeline

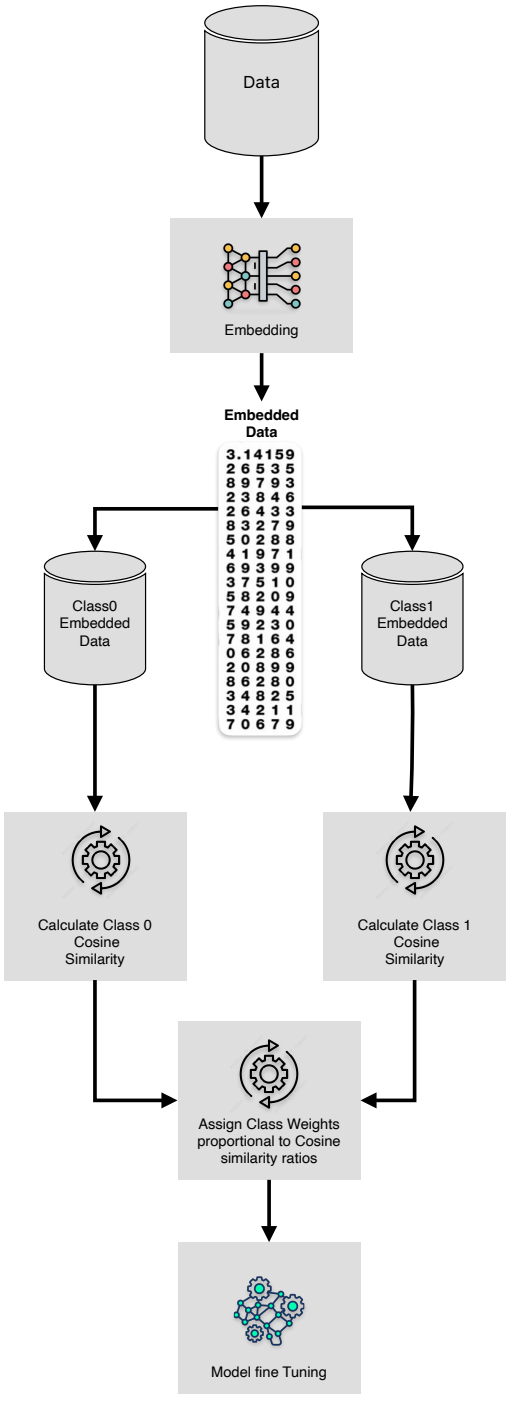


FIGURE 9: Finding the best class weights

Algorithm 1 finding the best class weights

```
for binary classification do
    Embed data using a pre-trained embedding model
    Split embedded data into two halves:
       class_0_data contains all data for class 0
       class_1_data contains all data for class 1
    for each class do
        Initialize total_cosine_similarity = 0
        for i = 1 to N − 1 do
            for j = i + 1 to N do
                total_cosine_similarity += cosine_similarity(data_i, data_j)
            end for
        end for
        if class == class_0 then
            class_0_total_cosine_similarity=total_cosine_similarity
        else
            class_1_total_cosine_similarity=total_cosine_similarity
        end if
    end for
    if class_0_total_cosine_similarity < class_1_total_cosine_similarity then
        class_0_weight = 1
        class_1_weight = class_0_total_cosine_similarity / class_1_total_cosine_similarity
    else
        class_1_weight = 1
        class_0_weight = class_1_total_cosine_similarity / class_0_total_cosine_similarity
    end if
    Use class_0_weight and class_1_weight for training of the embedding model
end for
```

G. Pipeline Summary and Insights

Figures 8 and 9 and Algorithm 1 successively show the system pipeline, pseudocode for computing class weights, and the suggested framework for optimal class weight determination.

This method has a major benefit in that it subtly takes into account the number of instances without explicitly considering them in the class weights computation. For example, assuming initially that class 1 requires a higher weight than class 0, let us have 100 instances for each class. Consequently, weights like class 0's weight: 0.5 and class 1's weight: 1. If we now add ten more records to class 1, one would assume that the rise in cases will cause class 1 to have a lesser weight than class 0. Still, with our method, this isn't always the case. The class weight will change depending on the characteristics of the extra records instead of depending solely on the rise in instance count.

Our method captures not just the instance count but also the similarity distribution inside each class since we rely on total cosine similarity inside every class to establish weights. Since the method balances weights depending on similarity rather than actual instance counts, the extra records for class 1 will likely close the weight difference between the classes. This approach guarantees a more complex and flexible class weighting scheme by essentially accounting for, for instance, the quantity without formally including it as a parameter.

For more clarification, Algorithm 1 outlines the step-by-step procedure for computing adaptive class weights that combine both class imbalance and intra-class semantic similarity. The process begins by dividing the dataset into separate classes: class 0 and class 1, and generating contextual embeddings for all instances using a pre-trained embedding model, such as XLM-Roberta-Large. These embeddings capture the semantic features of each sample, enabling the algorithm to assess the relationships between data points within the same class. By splitting the data into distinct classes, the algorithm ensures that the subsequent similarity calculations are performed independently for each class, allowing for a precise evaluation of intra-class characteristics.

Next, the algorithm calculates pairwise cosine similarity scores among all samples within each class to quantify semantic redundancy or diversity. This involves iterating through every possible pair of embeddings within a class and computing their cosine similarity, which measures the degree of relatedness between the samples in the embedding space. The scores are aggregated to produce a total intra-class similarity score for each class, reflecting the overall semantic

cohesion or variability within that class. For instance, a high total similarity score suggests that the class contains many semantically redundant samples, while a low score indicates greater diversity among the samples. This step is critical for understanding the internal structure of each class and informing the weight assignment process.

Finally, based on the aggregated similarity scores, class weights are computed using a normalization strategy. The class with the lower total similarity (indicating higher semantic diversity) is assigned a higher weight (defaulting to 1), encouraging the model to prioritize learning from its more varied examples. Conversely, the class with higher internal similarity receives a proportionally lower weight, reflecting its semantic redundancy and reduced informational contribution. If, for instance, class 0 has a total similarity score half that of class 1, class 1's weight would be set to 0.5, while class 0's weight remains 1. These weights are then used during the training of the embedding model to balance the influence of each class, ultimately improving the model's ability to generalize across diverse and imbalanced data.

Figure 9 visually reinforces this concept by showing how the aggregated intra-class similarity is translated into adaptive weights. It illustrates that the assigned weights do not solely depend on the number of samples but rather adapt according to the semantic nature of the added instances. This behavior enables our method to respond to structural changes in class content without depending strictly on frequency.

Together, Algorithm 1 and Figure 9 represent the core mechanics of our proposed class-weighting strategy. By embedding them within the Methodology section and expanding their discussion, we aim to provide a clear and replicable understanding of how our approach addresses both frequency imbalance and semantic variation.

VI. RESULTS AND DISCUSSION

All experiments were conducted on an HP Z6 G4 Workstation equipped with a 12-core Intel Xeon W-3235 processor running at 3.3 GHz, 128 GB of RAM, and an NVIDIA Quadro RTX 6000 GPU with 24 GB of VRAM. This setup provided sufficient computational capacity to train large transformer-based models such as XLM-Roberta-Large and to compute intra-class cosine similarities for each dataset. On average, training each model required approximately 2.5 to 3 hours, depending on the dataset size, batch size, and the number of epochs (40 in our case). The environment did not use multi-GPU training or distributed computing. All experiments were implemented using the PyTorch and the Hugging Face Transformers libraries, ensuring efficient GPU utilization and reproducibility.

We evaluate the performance of our classification models using the Accuracy metric, which measures the proportion of correctly classified samples over the total number of samples. It is computed as:

$$Accuracy = \frac{Number\ of\ Correct\ Predications}{Total\ Number\ of\ Predications}$$

This metric is appropriate for our binary and multi-class classification tasks, providing a direct and interpretable measure of model performance.

Using our method on the MSRPC dataset produced ideal class weights: class 1 (paragraphs) set to 1 and class 0 (not-paragraphs) set to 0.7980. These changed weights attained a 0.8325 test accuracy.

Training with equal class weights—that is, one against another—generated a somewhat lower test accuracy of 0.8175. The 0.015, or 1.5%, difference highlights the advantage of our intra-class similarity-based weighing technique above conventional equal-weight systems.

Table 3 shows how our results match previous studies, therefore stressing the effectiveness of this new weighting method in raising classification accuracy.

Extending our technique to the Semantic Question Similarity in Arabic dataset, part of the Solutions for Under-Resourced Languages (NSURL) project, based on the encouraging outcomes from the MSRPC dataset. This extra experiment sought to confirm the generalizability and strength of our approach even more. Regarding the NSURL dataset, ideal class weights were found as follows:

- Class 0 (not comparable): 0.6686
- Class 1: (like) 1

With these customized weights, we attained 0.9693 as the testing accuracy. Training with equal weights (1 vs. 1) produced a rather lower accuracy of 0.9604. Given the already excellent baseline accuracy, the difference, just 0.0089, or 0.89%, is noteworthy even in tiny margins and shows that our method improves performance even in these margins.

Table 4 offers a thorough comparison of our method against past performance, therefore highlighting the effectiveness of intra-class similarity-based weighting in producing outstanding classification results.

TABLE 3: RESULTS OF THE PROPOSED MODEL COMPARED WITH PREVIOUS WORKS ON THE MSRPC DATASET.

Study	Accuracy %
Corley and Mihalcea [43]	71.5
Mihalcea [44]	70.3
Vasile Rus [45]	70.61
Guo and Diab [46]	71.51
Lee and Cheah [47]	71.1
Zhang and Patrick [48]	71.9
Salakoski et al. [49]	76.64
Wan et al. [50]	75
Qiu et al. [51]	72

Malakasiotis [52]	76.17
Das and Smith [53]	80.41
Ul-Qayyum and Altaf [54]	74.67
Bu et al. [55]	76.3
Yin and Schütze [56]	78.7
Eyecioglu and Keller [57]	74.2
Marchenko et al. [58]	77.86
Madnani et al. [59]	77.4
Socher et al. [60]	76.8
Hu et al. [61]	69.9
Yin and Schütze [62]	78.1
He et al.[63]	78.60
Huang et al. [64]	73.3
Wang et al. [65]	78.4
Shen et al. [67]	80.92
Proposed Approach	**83.25**

Across two benchmark datasets, the Microsoft Research Paragraph Corpus (MSRPC) and the Semantic Question Similarity in Arabic (NSURL), our method produced outstanding performance. It routinely exceeded conventional and modern techniques, including semantic similarity techniques, classical feature extraction, machine learning models, and deep learning architectures. By capturing not just class imbalance but also intra-class similarity, our approach provided better accuracy than our baseline and previous attempts, incorporating equal class weighting and other tested

TABLE 4: PROPOSED MODEL RESULTS COMPARED WITH PREVIOUS WORKS REPORTED IN [66] ON THE NSURL DATASET

Team Name	Accuracy %
Eyad Sibai	71.434
AtyNegar	82.583
Speech Translation	82.698
Heza	85.736
Dan Ofer	89.465
Ayat Abedalla	91.311
Onekaggler	94.809
Tha3aroon	94.848
The Inception	95.924
Proposed Approach	**96.931**

TABLE 5: COMPARATIVE ACCURACY ACROSS TRAINING STRATEGIES

Method	MSRPC	NSURL
Baseline (Unweighted)	81.75	96.04
Frequency-Based	82.50	96.75
Proposed (Intra-Class)	**83.25**	**96.93**

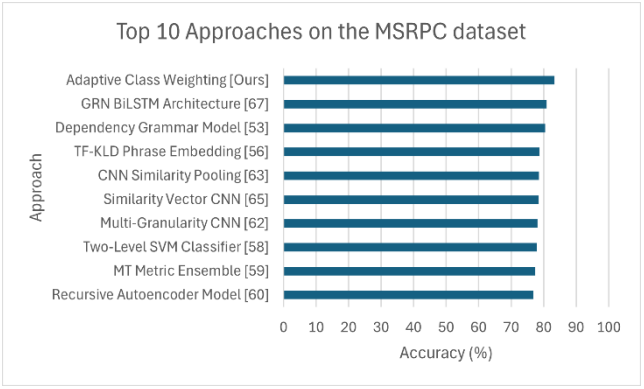


FIGURE 10: Top 10 Approaches on the MSRPC dataset

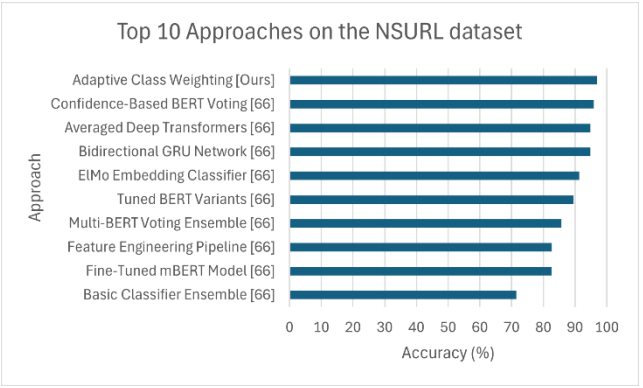


FIGURE 11: Top 10 Approaches on the NSURL dataset

weight combinations. By balancing the classes on both the instance frequency and intrinsic similarity levels, this intra-class similarity-based weighing has been shown to be successful in enhancing prediction dependability—an improvement not usually addressed in traditional weighting approaches.

There are trade-offs, even if this method clearly has advantages. Computing the cosine similarity across records inside every class takes extra computing time, which could increase greatly in bigger datasets. While current GPUs and parallel processing can assist in reducing some of this load, it is still a consideration, particularly in settings with limited resources. Another level of complication arises from the choice of embedding model and its tuning values. While choosing an ideal model for embedding and fine-tuning is iterative and data-dependent, most machine learning projects nowadays share this difficulty rather than being specific to our approach.

Our method is a strong substitute for similar activities in many fields since it can consider both intra-class similarity and record imbalance. More investigation could look at ways to simplify the computational needs by perhaps using more effective similarity metrics or adaptive sampling methods, hence enabling larger applications without appreciable processing costs.

To conclude, we conducted a comparative evaluation across three training configurations: (1) no class weighting (baseline), (2) frequency-based class weighting, and (3) our adaptive class weighting approach based on intra-class similarity.

On the MSRPC dataset, training without class weighting resulted in a test accuracy of 81.75%. Frequency-based weighting improved the performance slightly, but our proposed method achieved a higher test accuracy of 83.25%, representing a 1.5% improvement over equal-weight training. This improvement highlights the effectiveness of incorporating intra-class similarity, especially in datasets that exhibit both label imbalance and semantic redundancy.

On the NSURL dataset, the baseline (unweighted) configuration achieved 96.04% accuracy, whereas our proposed method reached 96.93%, marking a 0.89% improvement. Despite the already high baseline, this gain remains notable due to the competitive nature of this benchmark and the difficulty in achieving measurable improvements at this performance level.

Furthermore, as shown in Tables 3 and 4, our method consistently outperformed previously reported models on both datasets. On MSRPC, our accuracy of 83.25% exceeded results from several strong models, including Shen et al. (80.92%) [67], Das and Smith (80.41%) [50], and Yin and Schütze (78.70%) [56]. On NSURL, our approach surpassed the highest previously reported accuracy of 95.92% by the "The Inception" team, confirming its superiority.

Table 5 presents test accuracy for both the MSRPC and NSURL datasets across these configurations. Our method consistently outperforms the others, with accuracy gains ranging from 0.89% to 1.5%, even when the baseline performance is already high.

To assess the statistical significance of the proposed approach against the previous work shown in Tables 3 and 4, we found that only the classification accuracy was reported for the MSRPC and NSURL datasets, respectively. However, accuracy alone is insufficient to assess the statistical significance of performance differences compared to our proposal. Therefore, we evaluated statistical significance only between our proposed method and two baseline classifiers that we implemented ourselves, as their complete prediction results were available. The baseline classifiers are Baseline-Unweighted and Frequency-Based.

As shown in Table 5, the proposed approach outperformed both baselines in terms of accuracy. To determine whether these improvements were statistically significant, we applied McNemar's test to compare the predictions of the proposed approach with each baseline for both datasets (MSRPC and NSURL). Table 6 reports the resulting p-values. In all cases, the p-values were ≤ 0.05, indicating that the differences in performance between the proposed approach and the baselines are statistically significant. These results provide strong evidence for the effectiveness and reliability of our intra-class similarity–based weighting approach.

TABLE 6. MCNEMAR'S TEST P-VALUES COMPARING THE PROPOSED APPROACH WITH THE TWO BASELINES ON THE MSRPC AND NSURL DATASETS.

Baseline classifier	p-value for MSRPC	p-value for NSURL
Baseline-Unweighted	0.024	0.0043
Frequency-Based	0.030	0.0050

Figures 10 and 11 summarize the top 10 performing models on the MSRPC and NSURL datasets, respectively, and confirm that our proposed method consistently achieves the highest accuracy among all previous works and challenge submissions.

The comparative results further show that our method outperforms a wide range of traditional and deep learning models. On the MSRPC dataset, it exceeds the performance of models such as Recursive Autoencoders, CNN-based architectures, and GRN-BiLSTM. On the NSURL dataset, it outperforms several competitive BERT-based ensemble systems and advanced neural architectures. The observed improvements of +2.33% (MSRPC) and +1.01% (NSURL) highlight the robustness and generalization ability of our intra-class similarity-based weighting strategy, especially in the presence of semantic redundancy and class imbalance.

These results validate that our model achieves state-of-the-art performance on both benchmarks. The improvements are primarily due to our adaptive weighting strategy, which assigns higher weights to semantically diverse classes and lower weights to redundant ones, thus promoting more balanced and generalizable learning across all class types.

The intra-class similarity calculation requires computing pairwise cosine similarities within each class, which introduces a time complexity of $O(n_c^2)$ per class, where n_c is the number of instances in class c. For large datasets, this can become a bottleneck. However, several strategies can be applied to mitigate this cost:

1. **Offline Preprocessing**: The intra-class similarity scores are static for a given dataset and do not need to be recomputed during inference. As such, they can be computed once offline during the training preparation phase and reused throughout training and evaluation.
2. **GPU Acceleration and Parallel Processing**: Computing cosine similarities is highly parallelizable and can be efficiently implemented using GPU-accelerated libraries such as PyTorch or TensorFlow. Batch-wise matrix operations (e.g., dot products followed by normalization) allow the

computation to scale linearly across classes when distributed properly.

3. **Sampling-Based Approximation**: For extremely large classes, a representative random sample can be selected to estimate intra-class similarity with high confidence. This reduces the quadratic overhead to linear or sub-quadratic levels while preserving the quality of the similarity estimation.
4. **Vector Quantization or Clustering**: Clustering embeddings into a small number of centroids (e.g., using K-means) per class and computing similarities among centroids instead of individual samples offers another approximation approach that can drastically reduce computation.
5. **Incremental or Online Updates**: For streaming or dynamic datasets, incremental statistics (e.g., moving average of similarity) can be maintained instead of full recomputation.

By applying these strategies, particularly offline computation, parallel GPU-based processing, and sampling, we significantly reduce the practical computational burden of our method, enabling scalability to large real-world datasets while maintaining the benefits of adaptive class weighting.

While the current study focuses on binary classification tasks in Arabic text, the proposed methodology is fully extensible to multi-class scenarios. In a multi-class setting with $C > 2$ classes, our intra-class similarity-based weighting framework can be generalized by computing a similarity matrix $S \in \mathbb{R}^{C \times 1}$, where each entry S_c represents the total intra-class cosine similarity for class c.
Let the set of samples be:

$$D = \{(x_i, y_i)\}_{i=1}^{N}, y_i \in \{0, 1, \dots, C-1\}$$

For each class c, we compute the total intra-class similarity by summing over all pairs of samples that belong to class c:

$$S_c = \sum_{(x_i, x_j) \in D_c} \cos\left(E(x_i), E\left(x_j\right)\right)$$

Where *E(x)* denotes the contextual embedding of sample *x*.
We then normalize these scores to derive adaptive class weights:

$$w_c = \frac{\min(S)}{S_c}, \forall_c \in \{0,1, \dots, C-1\}$$

This formulation ensures that classes with lower intra-class similarity (i.e., more semantic diversity) are given higher importance during training, thus enhancing generalization in complex multi-class classification tasks.
Future work will involve validating this extension on multi-class Arabic classification benchmarks, including topic categorization and intent classification datasets. We also plan to integrate inter-class similarity measures to better handle class overlap and refine the weight calculation by incorporating both intra-class variance and inter-class separation using metrics like Fisher Discriminant Ratio.

This generalization aligns with the increasing need for robust NLP models that handle nuanced, multi-class tasks in morphologically rich and low-resource languages like Arabic.

In summary, the proposed approach not only improves accuracy but also introduces a more principled and interpretable method for addressing class imbalance, especially valuable in NLP tasks where semantic variability within classes plays a critical role.

VII. CONCLUSION AND FUTURE WORK

With particular attention to intra-class similarity, we introduced a fresh method for class weight assignment designed to solve imbalances in datasets in this study. While conventional balancing methods may presume that an equal number of records for every class indicates dataset balance, this method ignores intra-class similarity variations, which may reduce model performance. We proposed a method of dynamically computing class weights using embedding models along with cosine similarity measures to close this gap. This approach not only considers the record count per class but also catches intra-class similarities, so offering a more complete balance in the training data.

Our work is especially pertinent to Natural Language Processing (NLP) problems in Arabic, where special difficulties include dialect variation, a large vocabulary of around 12 million unique terms, and a relative scarcity of annotated resources. We evaluated our method on two benchmark datasets: Semantic Question Similarity in Arabic (NSURL) and the Microsoft Research Paragraph Corpus (MSRPC). These datasets' identical objectives and defined criteria for comparison gave a framework for both exploration and validation. By means of substantial empirical experimentation, our approach clearly outperformed conventional approaches, including classical semantic and text similarity models, as well as other machine learning and deep learning architectures incorporating preprocessing techniques. Our method especially exceeded state-of-the-art results by achieving an accuracy of 83.25% on the MSRPC dataset and 96.93% on the Arabic question similarity dataset, thus confirming the effectiveness of our intra-class similarity balancing strategy.

Our results inspire more investigation on controlling dataset imbalances and extending this approach to multi-class classification issues, as well as across many languages and fields outside NLP, such as numerical data analysis and computer vision. We intend to investigate optimal embedding models in further work to lower computing time and modify the method for bigger, multi-dimensional datasets. Historically, we seek to examine its applicability to datasets in other languages and dialects as well as to expand to

unstructured data in multimedia contexts, perhaps enhancing strong balancing tactics across many machine learning applications.

ACKNOWLEDGMENT

This work is supported by the Deanship of Scientific Research, Islamic University of Madinah, Saudi Arabia.

CONFLICTS OF INTEREST

The authors declare no conflict of interest.

AUTHORS CONTRIBUTIONS

Methodology, Emad Nabil, Abdelrahman Nagib, Mena Hany, Safiullah Faizullah, and Wael Gomaa; Project administration, Emad Nabil; Software, Abdelrahman Nagib and Mena Hany; Supervision, Emad Nabil, Safiullah Faizullah, and Wael Gomaa; Writing – original draft, Abdelrahman Nagib and Mena Hany; Writing – review & editing, Emad Nabil, Abdelrahman Nagib, Mena Hany, Safiullah Faizullah and Wael Gomaa.

REFERENCES

[1] F. A. Saâdane, B. Gueni, and D. Nouvel, "Arabic natural language processing: An overview," *Journal of King Saud University-Computer and Information Sciences*, vol. 33, no. 5, pp. 497-507, 2021.

[2] M. Boudchiche and A. Mazroui, "A hybrid approach for Arabic lemmatization," *Int J Speech Technol*, vol. 22, no. 3, pp. 563–573, Sep. 2019, doi: 10.1007/s10772-018-9528-3.

[3] D. Namly, K. Bouzoubaa, A. El Jihad, and S. L. Aouragh, "Improving Arabic Lemmatization Through a Lemmas Database and a Machine-Learning Technique," in *Recent Advances in NLP: The Case of Arabic Language*, M. Abd Elaziz, M. A. A. Al-qaness, A. A. Ewees, and A. Dahou, Eds., Cham: Springer International Publishing, 2020, pp. 81–100. doi: 10.1007/978-3-030-34614-0_5.

[4] S. AL-Sarayreh, A. Mohamed, and K. Shaalan, "Challenges and Solutions for Arabic Natural Language Processing in Social Media," in *Business Intelligence and Information Technology*, A. E. Hassanien, D. Zheng, Z. Zhao, and Z. Fan, Eds., Singapore: Springer Nature, 2023, pp. 293–302. doi: 10.1007/978-981-99-3416-4_24.

[5] M. Abdul-Mageed, "NADI 2023: The fourth nuanced Arabic dialect identification shared task." 2023.

[6] S. Jamal, A. M. Kassem, O. Mohamed, and A. Ashraf, "On the Arabic dialects' identification: Overcoming challenges of geographical similarities between Arabic dialects and imbalanced datasets," in *Proceedings of the seventh Arabic natural language processing workshop (WANLP*, 2022, pp. 458–463.

[7] A. Waheed, B. Talafha, P. Sullivan, A. Elmadany, and M. Abdul-Mageed, "VoxArabica: A robust dialect-aware Arabic speech recognition system." 2023.

[8] W. H. Gomaa, A. E. Nagib, M. M. Saeed, A. Algarni, and E. Nabil, "Empowering short answer grading: Integrating transformer-based embeddings and BI-LSTM network," *Big Data and Cognitive Computing*, vol. 7, no. 3, p. 122, 2023.

[9] J. Younes, E. Souissi, H. Achour, and A. Ferchichi, "Language resources for Maghrebi Arabic dialects' NLP: A survey," *Language Resources and Evaluation*, vol. 54, pp. 1079-1142, 2020.

[10] K. Ghosh, "The class imbalance problem in deep learning," *Machine Learning*, vol. 113, no. 7, pp. 4845-4901, 2024.

[11] M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *Journal of Big Data*, vol. 6, no. 1, pp. 1-54, 2019.

[12] T. Huang, Q. She, and J. Zhang, "BoostingBERT: Integrating multi-class boosting into BERT for NLP tasks." 2020.

[13] M. A. Elaziz, M. A. A. Al-qaness, A. A. Ewees, and A. Dahou, *Recent advances in NLP: The case of Arabic language*. Springer Nature, 2019.

[14] E. Wardini, "Arabic computational linguistics: Potential, pitfalls and challenges," in *Natural language processing in artificial intelligence—NLPinAI 2021*, Springer, 2022, pp. 105–117.

[15] E. Saiegh-Haddad and R. Henkin-Roitfarb, "The structure of Arabic language and orthography," in *Handbook of Arabic literacy: Insights and perspectives*, Springer, 2014, pp. 3–28.

[16] A. G. Martínez, S. L. Hervás, D. Samy, C. G. Arques, and A. M. Sandoval, "Jabalín: A comprehensive computational model of modern standard Arabic verbal morphology based on traditional Arabic prosody," in *Systems and frameworks for computational morphology: Third international workshop, SFCM 2013*, Berlin, Germany: Springer, 2013, p. 3.

[17] A. Soudi, G. Neumann, and A. Bosch, "Arabic computational morphology: Knowledge-based and empirical methods," in *Arabic computational morphology: Knowledge-based and empirical methods*, Springer, 2007, pp. 3–14.

[18] A. R. Ali, M. A. Siddiqui, R. Algunaibet, and H. R. Ali, "A large and diverse Arabic corpus for language modeling," *Procedia Computer Science*, vol. 225, pp. 12-21, 2023.

[19] M. Hany, M. M. Saeed, R. R. Waly, A. E. Nagib, and W. H. Gomaa, "Enhancing textual relatedness assessment with combined transformers-embedding similarity techniques and machine learning regressors," in *2023 intelligent methods, systems, and applications (IMSA*, IEEE, 2023, pp. 13–18.

[20] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," *Electronic Markets*, vol. 31, no. 3, pp. 685-695, 2021.

[21] M. Maree and W. Shehada, "Optimizing curriculum vitae concordance: A comparative examination of classical machine learning algorithms and large language model architectures," *AI*, vol. 5, no. 3, pp. 1377-1390, 2024.

[22] I. H. Sarker, "Machine learning: Algorithms, real-world applications and research directions," *SN Computer Science*, vol. 2, no. 3, p. 160, 2021.

[23] A. E. Nagib, M. M. Saeed, S. F. El-Feky, and A. K. Mohamed, "Neural network with adaptive learning rate," in *2020 2nd novel intelligent and leading emerging sciences conference (NILES*, IEEE, 2020, pp. 544–548.

[24] A. E. Nagib, H. Mohamed, and A. Ashraf, "Comparative analysis of pre-trained CNN architectures for multi-class weather classification: A study on deep learning techniques," in *2023 intelligent methods, systems, and applications (IMSA*, IEEE, 2023, pp. 297–302.

[25] H. Mohamed, A. Ashraf, and A. E. Nagib, "Comparative study of Alzheimer's disease classification using transfer learning models," in *2023 intelligent methods, systems, and applications (IMSA*, IEEE, 2023, pp. 434–439.

[26] A. E. Nagib, M. M. Saeed, S. F. El-Feky, and A. K. Mohamed, "Hyperparameters optimization of deep convolutional neural network for detecting COVID-19 using differential evolution," in *Decision sciences for COVID-19: Learning through case studies*, Springer, 2022, pp. 305–325.

[27] A. Ashraf, A. E. Nagib, and H. Mohamed, "Enhancing breast cancer diagnosis with vision transformer-based ultrasound image classification," in *2023 5th novel intelligent and leading emerging sciences conference (NILES*, IEEE, 2023, pp. 161–165.

[28] H. Mohamed, A. E. Nagib, and M. Hany, "Comparative study of microscopic fungii classification using transfer learning models," in *2024 intelligent methods, systems, and applications (IMSA*, IEEE, 2024, pp. 87–92.

[29] B. Jang, M. Kim, G. Harerimana, S. Kang, and J. W. Kim, "Bi-LSTM model to increase accuracy in text classification: Combining Word2vec CNN and attention mechanism," *Applied Sciences*, vol. 10, no. 17, p. 5841, 2020.

[30] S. L. Ramaswamy and J. Chinnappan, "RecogNet-LSTM+ CNN: A hybrid network with attention mechanism for aspect categorization and sentiment classification," *Journal of Intelligent Information Systems*, vol. 58, no. 2, pp. 379-404, 2022.

[31] M. Saeed, R. R. Waly, M. Hany, A. E. Nagib, and W. H. Gomaa, "Improving credibility detection with combined transformers-based with machine learning and deep learning techniques," in *2023 5th novel intelligent and leading emerging sciences conference (NILES*, IEEE, 2023, pp. 300–305.

[32] R. R. Waly, M. M. Saeed, M. Hany, A. E. Nagib, and W. H. Gomaa, "Empowering topic classification accuracy by integrating transformer-based with machine learning and deep learning techniques," in *2023 intelligent methods, systems, and applications (IMSA*, IEEE, 2023, pp. 19–25.

[33] S. Minaee, E. Azimi, and A. Abdolrashidi, "Deep-sentiment: Sentiment analysis using ensemble of CNN and Bi-LSTM models." 2019.

[34] 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, 2021.

[35] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263-1284, 2009.

[36] A. Fernández, S. Garcia, F. Herrera, and N. V. Chawla, "SMOTE for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary," *Journal of Artificial Intelligence Research*, vol. 61, pp. 863-905, 2018.

[37] T. T. Khoei, H. O. Slimane, and N. Kaabouch, "Deep learning: Systematic review, models, challenges, and research directions," *Neural Computing and Applications*, vol. 35, no. 31, pp. 23103-23124, 2023.

[38] B. Krawczyk, M. Koziarski, and M. Woźniak, "Radial-Based Oversampling for Multiclass Imbalanced Data Classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 8, pp. 2818-2831, Aug. 2020.

[39] S. Wang and X. Yao, "Multiclass Imbalance Problems: Analysis and Potential Solutions," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics*, vol. 42, no. 4, pp. 1119-1130, Aug. 2012.

[40] A. Y. Taha, S. Tiun, A. H. Abd Rahman, and A. Sabah, "Multilabel over-sampling and under-sampling with class alignment for imbalanced multilabel text classification," *Journal of Information and Communication Technology*, vol. 20, no. 3, pp. 423–456, 2021.

[41] M. Rafi-Ur-Rashid, M. Mahbub, and M. A. Adnan, "Breaking the curse of class imbalance: Bangla text classification," *Transactions on Asian and Low-Resource Language Information Processing*, vol. 21, no. 5, pp. 1–21, 2022.

[42] S. Shaikh, S. M. Daudpota, A. S. Imran, and Z. Kastrati, "Towards improved classification accuracy on highly imbalanced text dataset using deep neural language models," *Applied Sciences*, vol. 11, no. 2, p. 869, 2021.

[43] C. D. Corley and R. Mihalcea, "Measuring the semantic similarity of texts," in *Proceedings of the ACL workshop on empirical modeling of semantic equivalence and entailment*, 2005, pp. 13–18.

[44] R. Mihalcea, C. Corley, and C. Strapparava, "Corpus-based and knowledge-based measures of text semantic similarity," in *Proceedings of the AAAI conference on artificial intelligence*, 2006, pp. 775–780.

[45] V. Rus, P. M. McCarthy, M. C. Lintean, D. S. McNamara, and A. C. Graesser, "Paraphrase identification with lexico-syntactic graph subsumption," in *Proceedings of FLAIRS*, 2008, pp. 201–206.

[46] W. Guo and M. Diab, "Modeling sentences in the latent space," in *Proceedings of the 50th annual meeting of the association for computational linguistics (volume 1*, Long papers, 2012, pp. 864–872.

[47] J. C. Lee and Y.-N. Cheah, "Paraphrase detection using semantic relatedness based on Synset Shortest Path in WordNet," in *2016 international conference on advanced informatics: Concepts, theory and application (ICAICTA*, IEEE, 2016, pp. 1–5.

[48] Y. Zhang and J. Patrick, "Paraphrase identification by text canonicalization," in *Proceedings of the Australasian language technology workshop 2005*, 2005, pp. 160–166.

[49] F. Ginter and S. Pahikkala, *Advances in natural language processing*. Springer, 2006.

[50] S. Wan, M. Dras, R. Dale, and C. Paris, "Using dependency-based features to take the 'para-farce' out of paraphrase," in *Proceedings of the Australasian language technology workshop 2006*, 2006, pp. 131–138.

[51] L. Qiu, M.-Y. Kan, and T.-S. Chua, "Paraphrase recognition via dissimilarity significance classification," in *Proceedings of the 2006 conference on empirical methods in natural language processing*, 2006, pp. 18–26.

[52] P. Malakasiotis, "Paraphrase recognition using machine learning to combine similarity measures," in *Proceedings of the ACL-IJCNLP 2009 student research workshop*, 2009, pp. 27–35.

[53] D. Das and N. A. Smith, "Paraphrase identification as probabilistic quasi-synchronous recognition," in *Proceedings of the joint conference of the 47th annual meeting of the ACL and the 4th international joint conference on natural language processing of the AFNLP*, 2009, pp. 468–476.

[54] Z. Ul-Qayyum and W. Altaf, "Paraphrase identification using semantic heuristic features," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 4, no. 22, pp. 4894-4904, 2012.

[55] F. Bu, H. Li, and X. Zhu, "String re-writing kernel," in *Proceedings of the 50th annual meeting of the association for computational linguistics (volume 1*, Long papers, 2012, pp. 449–458.

[56] W. Yin and H. Schütze, "Discriminative phrase embedding for paraphrase identification." 2016.

[57] A. Eyecioglu and B. Keller, "Knowledge-lean paraphrase identification using character-based features," *Artificial intelligence and natural language: 6th conference, AINL 2017, St*. Springer, Petersburg, Russia, pp. 257–276, Sep. 20, 2017.

[58] O. Marchenko, A. Anisimov, A. Nykonenko, T. Rossada, and E. Melnikov, "Machine learning method for paraphrase identification," in *Flexible query answering systems: 12th international conference, FQAS 2017*, London, UK: Springer, Jun. 2017, p. 12.

[59] N. Madnani, J. Tetreault, and M. Chodorow, "Re-examining machine translation metrics for paraphrase identification," in *Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies*, 2012, pp. 182–190.

[60] R. Socher, E. Huang, J. Pennin, C. D. Manning, and A. Ng, "Dynamic pooling and unfolding recursive autoencoders for paraphrase detection," in *Advances in neural information processing systems*, vol. 24, 2011.

[61] B. Hu, Z. Lu, H. Li, and Q. Chen, "Convolutional neural network architectures for matching natural language sentences," in *Advances in neural information processing systems*, vol. 27, 2014.

[62] W. Yin and H. Schütze, "Convolutional neural network for paraphrase identification," in *Proceedings of the 2015 conference of the North American chapter of the association for computational linguistics: Human language technologies*, 2015, pp. 901–911.

[63] H. He, K. Gimpel, and J. Lin, "Multi-perspective sentence similarity modeling with convolutional neural networks," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015, pp. 1576–1586.

[64] J. Huang, D. Ji, S. Yao, and W. Huang, "Character-aware convolutional neural networks for paraphrase identification," *Neural information processing: 23rd international conference, ICONIP 2016*. Springer, Kyoto, Japan, pp. 177–184, Oct. 16, 2016.

[65] Z. Wang, H. Mi, and A. Ittycheriah, "Sentence similarity learning by lexical decomposition and composition." 2016.

[66] H. Seelawi, A. Mustafa, H. Al-Bataineh, W. Farhan, and H. T. Al-Natsheh, "Nsurl-2019 shared task 8: Semantic question similarity in Arabic." 2019.

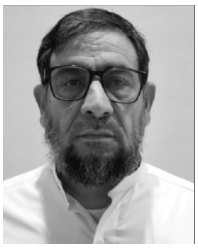
[67] Y. Shen, J. Chen, and X. Huang, "Bidirectional long short-term memory with gated relevance network for paraphrase identification," in *Natural language understanding and intelligent applications: 5th CCF conference on natural language processing and Chinese computing, NLPCC 2016, and 24th international conference on computer processing of oriental languages, ICCPOL 2016*, Kunming, China: Springer, Dec. 2016, p. 24.

[68] B. Dolan and C. Brockett, "Automatically constructing a corpus of sentential paraphrases," in *Third International Workshop on Paraphrasing (IWP2005*, 2005.

Emad Nabil: Received a B.Sc. degree in Computer Science (2004), an M.Sc. degree (2008), and a Ph.D. degree (2012) in Artificial Intelligence from the Faculty of Computers and Artificial Intelligence, Cairo University. Currently serves as a professor at both the Islamic University of Madinah, Saudi Arabia, and Cairo University, Egypt. Co-founder of the AI Center at the Islamic University of Madinah and consultant to the Decision Support Center at the same institution. Actively contributes as a reviewer for numerous academic journals and has supervised multiple master's and Ph.D. students. Research interests include Machine Learning, Deep Learning, Generative AI, Natural Language Processing, Soft Computing, and Health Informatics.

ABDELRAHMAN EZZELDIN NAGIB Born in 1999 in Egypt, with a bachelor's degree with honors in computer science in 2018 from MSA University in Egypt. He has experience in the A.I. field in natural language processing and Computer Vision. Worked in the A.I. field for over 5 years. Worked at Meet Platforms in Egypt for 4 years in NLP and Computer vision Worked at MSA in university Egypt as a teaching assistant and research for 3 years Worked at DigitalHub in Egypt as an A.I. instructor for deep learning and machine learning for 1 year A teaching assistant and research assistant in the field of computer science and deep learning with expertise and interests in various domains in machine learning such as Natural Language Processing, Computer Vision and Meta-Heuristic optimization for deep learning. Published a paper titled Empowering short answer grading: Integrating transformer-based embeddings and BI-LSTM network. Published a paper titled Hyperparameters optimization of deep convolutional neural network for detecting COVID-19 using differential evolution Published a paper titled Neural network with adaptive learning rate

MENA HANY is an AI Engineer, Data Scientist, and academic researcher with expertise in machine learning, natural language processing, and data science. He is currently pursuing a Master's in Computer Science at King Fahd University of Petroleum & Minerals (KFUPM) and works as an AI Engineer at Borne, developing and optimizing AI-driven solutions. In addition to his industry role, he serves as a Teaching Assistant at Modern Sciences and Arts University and conducts research in deep learning and NLP at the Islamic University of Madinah. He has previously worked as a Data Scientist at BuduCloud, focusing on predictive modeling and data analysis. holds dual bachelor's degrees in Computer Science from the University of Greenwich and October University for Modern Sciences & Arts. With strong technical and leadership skills, he is dedicated to advancing AI research and practical applications.



Safiullah Faizullah (Senior Member, IEEE) received the B.S. and M.S. degrees in information and computer science from the King Fahd University of Petroleum and Minerals (KFUPM), Dhahran, Saudi Arabia, in 1991 and 1994, respectively, and the M.S., M.Phil., and Ph.D. degrees in computer science from Rutgers University, New Brunswick, NJ, USA, in 2000, 2001, and 2002, respectively. Since July 1999, he has been with Hewlett-Packard Enterprise. He was with Isra University. Since July 2013, he has also been a Visiting Research Professor with the Department of Computer Science, at Rutgers University. He is currently a Full Professor of computer science with the Faculty of Computer and Information Systems at the Islamic University of Madinah. He has been teaching graduate-level courses on the fundamentals of software testing and cloud computing. He has authored over 25 journals and conference papers. His research interests include computer networks, cloud computing, mobile computing, wireless networks, distributed, graph theory, enterprise systems, and software testing. He is the Chair of the Curriculum Committee, the High-Performance Computing and Networking Research Group, and the Graduate Studies Program Application Committee, and

an Advisor to the Dean and the IT Deanship Dean. He is a member of PMI and ACM.

Wael Hassan Gomaa is an Associate Professor at Beni-Suef University, Faculty of Computers and Artificial Intelligence. He earned his PhD in Computer Science from Cairo University, Egypt, in 2014, with a specialization in Natural Language Processing (NLP). His research interests encompass AI, Large Language Models (LLMs), NLP, Machine Learning (ML), Deep Learning (DL), and Text Mining. He has authored over 25 papers published in high-ranked journals and works as a researcher and consultant for various NLP projects in different companies and ministries. Additionally, he has extensive experience in teaching, supervising research, and guiding graduation projects at universities in Egypt and Saudi Arabia (KSA).