applied sciences

MDPI

Article

Arabic Emotion Recognition in Low-Resource Settings: A Novel Diverse Model Stacking Ensemble with Self-Training

Maha Jarallah Althobaiti

Department of Computer Science, College of Computers and Information Technology, Taif University, Taif 21944, Saudi Arabia; maha.j@tu.edu.sa

Abstract: Emotion recognition is a vital task within Natural Language Processing (NLP) that involves automatically identifying emotions from text. As the need for specialized and nuanced emotion recognition models increases, the challenge of fine-grained emotion recognition with limited labeled data becomes prominent. Moreover, emotion recognition for some languages, such as Arabic, is a challenging task due to the limited availability of labeled data. This scarcity exists in both size and the granularity of emotions. Our research introduces a novel framework for low-resource fine-grained emotion recognition, which uses an iterative process that integrates a stacking ensemble of diverse base models and self-training. The base models employ different learning paradigms, including zero-shot classification, few-shot methods, machine learning algorithms, and transfer learning. Our proposed method eliminates the need for a large labeled dataset to initiate the training process by gradually generating labeled data through iterations. During our experiments, we evaluated the performance of each base model and our proposed method in low-resource scenarios. Our experimental findings indicate our approach outperforms the individual performance of each base model. It also outperforms the state-of-the-art Arabic emotion recognition models in the literature, achieving a weighted average F1-score equal to 83.19% and 72.12% when tested on the AETD and ArPanEmo benchmark datasets, respectively.

Keywords: fine-grained emotion recognition; limited labeled data; Arabic emotion recognition; stacking ensemble; self-training; few-shot learning; zero-shot classification

check for updates

Citation: Althobaiti, M.J. Arabic Emotion Recognition in Low-Resource Settings: A Novel Diverse Model Stacking Ensemble with Self-Training. *Appl. Sci.* **2023**, *13*, 12772. https://doi.org/10.3390/app132312772

Academic Editors: Douglas O'Shaughnessy and Bo Wang

Received: 24 September 2023
Revised: 19 November 2023
Accepted: 26 November 2023
Published: 28 November 2023

1. Introduction

Natural language processing is gaining momentum for its ability to shape interactions between humans and machines by automatically analyzing and comprehending human languages. Emotion recognition, a vital task in NLP, entails the automatic identification of human emotions from text [1]. This has a wide range of applications across industries, from healthcare and education to automotive industry and security. As the necessity for more specialized and granular emotion recognition increases, the challenge of low-resource, fine-grained emotion recognition becomes increasingly significant. Fine-grained emotion recognition is the process of automatically identifying emotions in textual data at a fine granularity level [2,3]. Unlike general emotion recognition, which categorizes emotions into broad classes such as "happiness", "anger", "sadness", and "fear", fine-grained emotion recognition include variations within these broader emotion classes. For example, instead of simply categorizing a given text as the "happiness" emotion, fine-grained emotion recognition distinguishes between emotions such as "joy", "satisfaction", "love", and "happiness".

Early techniques utilized for emotion recognition relied on manually engineered features, rule-based methods, and traditional Machine Learning (ML) algorithms [4–6]. These conventional methods require human intervention to create labeled data or to determine the rules and therefore can be time-consuming and may not generalize well across different domains. However, with the advancement of ML techniques, particularly deep learning models, emotion recognition has evolved [7–11].

Deep Learning (DL) models, such as Convolutional Neural Networks (CNNs) [12], Recurrent Neural Networks (RNNs) [13], and more recently Transformer-based models, such as Bidirectional Encoder Representations from Transformers (BERT) [14] and Generative Pretrained Transformer (GPT) [15], have shown remarkable performance in various text classification tasks, including emotion recognition [16–20]. These models can automatically learn features and representations from raw texts, avoiding the need for hand-engineered features.

Training a model for fine-grained emotion recognition requires a substantial amount of labeled data. However, the data can be limited for specialized domains or some granular emotion categories. Moreover, fine-grained classification often entails a large number of classes, making it challenging to maintain balanced datasets across classes [21].

The challenges faced by emotion recognition models, such as handling granular emotions, the requirement for a substantial amount of labeled data, and performing cross-domain emotion recognition, are applicable to many languages, including Arabic and English. However, when comparing Arabic to English, research on Arabic emotion recognition lacks annotated datasets. A recent survey on Arabic emotion recognition [22] revealed the limited availability of labeled datasets and emotion lexicons for Arabic. This challenge is intensified by the diversity of Arabic varieties in use today, including Modern Standard Arabic (MSA) and various Arabic dialects, which further complicates the task of creating labeled data for different Arabic varieties.

In this paper, we propose a novel method that addresses the challenges of low-resource fine-grained Arabic emotion recognition. Central to our methodology is an iterative framework that integrates a stacking ensemble of diverse models and self-training. The ensemble model, also known as the meta-learner or stacker, takes the predictions generated by the base models as input features and learns to combine them to make the final prediction. The base models include machine learning models, few-shot models, BERT-based model, and a zero-shot classification model. On the other hand, the base models and self-training with majority voting are utilized to carefully and gradually increase the amount of labeled data in each iteration. We utilized two different benchmark datasets for Arabic emotion recognition to evaluate our method and to compare it with other suggested methods in the literature. We can summarize the contributions of our paper as follows:

- We propose a novel method for low-resource, fine-grained Arabic emotion recognition integrating a stacking ensemble of diverse models and self-training.
- We investigate various methods (ML algorithms, few-shot methods, transfer learning, and a zero-shot classification method) for fine-grained, low-resource scenarios in Arabic emotion recognition.
- We examine the use of a stacking ensemble of base models employing diverse learning paradigms for Arabic emotion recognition.
- We compare the performance of our proposed method for Arabic emotion recognition with the state-of-the-art methods in the literature.

Our proposed method is language-agnostic and does not incorporate modules dedicated to language-specific processing. It can be applied to any language in low-resource settings. However, we utilize pretrained language models, and their availability and diversity are critical for the effectiveness of our approach. Therefore, when considering the adoption of our proposed method, the accessibility of various large language models is vital. It is essential to recognize that the extent of resource scarcity may differ from one language to another, and this factor should be taken into account.

The remainder of this paper is organized as follows. Section 2 presents an overview of the related works and available datasets for Arabic emotion recognition. In Section 3, the general framework and architecture of our novel approach to fine-grained Arabic emotion recognition are outlined. Sections 4 and 5 provide detailed explanations of each component of our proposed method, including the diverse model stacking ensemble, and the self-training process. Experiments, including the utilized dataset, data preprocessing, evaluation metrics, and the results, are presented in Section 6, followed by a discussion in Section 7. Finally, our conclusions and insights are presented in Section 8.

2. Related Work

This section begins by providing a summary and discussion of the latest research in the field of fine-grained emotion recognition in general. Subsequently, it explores related works in the literature that suggest various methods for Arabic emotion recognition, ensuring clarity for later comparisons with our proposed method. Lastly, we review the available annotated Arabic emotion recognition datasets, considering factors such as their size, the Arabic varieties studied, the emotion taxonomy, and the methods employed to collect them.

2.1. Fine-Grained Emotion Recognition

Fine-grained emotion recognition in text has obtained significant attention in recent years, with ongoing research being focused on developing accurate and effective methods. Deep learning and Pretrained Language Models (PLMs) have been extensively explored in recent research studies [23–25], as these models provide a robust foundation for fine-grained emotion recognition by capturing intricate patterns and contextual cues. The study by Zygadło et al. [26] employed various classifiers, including Naive Bayes, Support Vector Machine (SVM), and BERT, for 9-class emotion recognition in both English and Polish. The BERT-based model outperformed the conventional ML methods, achieving an accuracy of nearly 80% for emotion recognition in English and approximately 75% in Polish.

The study conducted by [27] proposed a deep learning-based model for 7-class emotion recognition during the COVID-19 pandemic. The model relied on combining a Long Short-Term Memory (LSTM) model and the BERT model. Specifically, the model consists of BERT as the top layer, followed by a dropout layer. The output is then fed into an LSTM layer, followed by several dropout and dense layers. The model was evaluated on a benchmark dataset, achieving an accuracy and F1-score of 83% and 72%, respectively. Adoma et al. [28] explored the efficiency of BERT, DistilBERT, Generalized Auto-regression Pretraining for Language Understanding (XLNet) [29], and Robustly Optimized BERT Pretraining Approach (RoBERTa) [30] models in recognizing emotions from texts. Using the same hyperparameters, the accuracies of the models, in decreasing order, were 74.31%, 72.99%, 70.09%, and 66.93% for RoBERTa, XLNet, BERT, and DistilBERT, respectively.

A recent study by Kumar et al. [31] introduced a novel dual-channel method for emotion recognition in text. The architecture of the proposed method comprises several modules. Firstly, the embedding module utilizes the BERT model in order to extract textual features from input sentences in the form of embedding vectors. Subsequently, the dual-channel module takes the output of the embedding module as input and transfers it to two network channels, which consist of a CNN and bidirectional LSTM (BiLSTM). The embedding vectors from both channels are concatenated and then fed into the next module, which is the emotion classification module. This module learns and projects the emotion embeddings onto a hyperplane, forming clusters. The proposed method was examined using four different emotion recognition datasets, consisting of examples categorized into emotion classes ranging from 5 to 7 emotions. The method showed consistent performance across the four different datasets, with accuracy ranging from 72.89% to 80.67% and F1-scores ranging from 71% to 83%.

Fine-grained emotion recognition and its potential applications often require taxonomies that encompass a considerable number of emotions, including nuanced states. This need for detailed emotion categories is vital for a deeper understanding of emotional content. Consequently, the development of more comprehensive taxonomies has recently attracted research attention. Demszky et al. [23] proposed a new taxonomy by introducing the GoEmotions dataset, which contains 58K English Reddit comments labeled as 27 emotion categories or neutral. The study by Sosea et al. [24] created an emotion dataset from an online health community, annotated with 8 fine-grained emotions. Indeed, the field of fine-grained emotion recognition has witnessed the emergence of several datasets that vary in size, domain, and taxonomy [32–34]. In general, emotion taxonomies and specialized labeled data continue to evolve and expand, paralleling the increasing potential applications for emotion recognition.

2.2. Arabic Emotion Recognition

Early research studies on Arabic emotion recognition relied on lexicons [35–37] and machine learning algorithms [38–40]. In these studies, emotion lexicons and machine learning algorithms, such as Naive Bayes, Sequential Minimal Optimization, and SVM [41], were employed to develop emotion classifiers.

Mohammad et al. [42]organized SemEval-2018 Task 1: Affect in Tweets, which included a subtask for multi-label emotion classification in Arabic. The study by Badaro et al. [43], which secured the top rank on the leaderboard, explored the use of three learning algorithms: Ridge Classification, Random Forests, SVM, and an ensemble of the three. They examined a set of features, including sentiment lexicons, n-grams, AraVec embeddings [44], and FastText embeddings [45]. The SVM classifier outperformed other models, while AraVec word embeddings outperformed other features, achieving accuracy rates of 48.9%, a micro average F1-score of 61.8%, and a macro average F1-score of 46.10%.

Mulki et al. [40] conducted binary classification using a one-vs.-all SVM strategy to address the multi-label emotion classification subtask in the SemEval-2018 challenge. In this approach, a set of binary SVM classifiers, equal to the number of emotion labels, was built in parallel to recognize the emotions in the given texts. Their proposed system achieved an accuracy of 46.5%, a micro average F1-score of 59.7%, and a macro average F1-score of 44.6%.

Deep learning models and various neural network architectures were explored for the identification of human emotions in Arabic text. Abdullah and Shaikh [46] addressed the multi-label Arabic emotion classification subtask in SemEval-2018 [42] as a binary classification problem. Their proposed neural network consisted of three hidden layers with a Rectified Linear Unit (RELU) activation function, and the output layer comprised a single sigmoid neuron predicting the intensity of the emotion on a scale from 0 to 1. This real-valued number, ranging from 0 to 1, was subsequently normalized to either 1 (indicating the presence of an emotion) if it exceeded 0.5 or 0 (indicating no emotion) if it fell below 0.5. They reported that their system achieved fourth place in the emotion classification subtask, with an accuracy rate of 44.60%.

The study by Alswaidan et al. [47] proposed three models for Arabic emotion recognition in text: the Human-Engineered Feature-based (HEF) model, the Deep Feature-based (DF) model, and a hybrid model called HEF+DF. The HEF model employed various features, including domain-specific, stylistic, lexical, syntactic, and semantic features. Deep neural networks (DNNs) with ReLU and sigmoid activation functions were utilized in this model. The DF model, on the other hand, employed LSTM and Gated Recurrent Unit (GRU) deep models along with pretrained word embeddings, which included emoji2vec [48], AraVec [44], Glove [49], and FastText [45,50]. The study found that the hybrid of both models (HEF + DF) outperformed each individual model when evaluated on the Arabic Emotions Twitter Dataset (AETD). The HEF+DF model achieved an accuracy of 71.8%, with a weighted-average F1-score, precision, and recall of 71.8%, 72.2%, and 71.8%, respectively.

Similar to other languages, PLMs have recently garnered significant attention due to their valuable contributions to state-of-the-art results in many NLP tasks. They serve as effective transfer learning techniques that can be fine-tuned on limited labeled data and produce excellent results for various downstream tasks [51–56]. Several studies have applied PLMs and transfer learning to Arabic emotion recognition in text [55–57].

The study by Abdul-Mageed et al. [56] introduced ARBERT and MARBERT, two PLMs specifically designed for Arabic language understanding. MARBERT was pretrained on social media data, which included 1 billion Arabic tweets. In contrast, ARBERT was mainly pretrained on MSA datasets. These PLMs were evaluated on various social meaning tasks, including emotion recognition. The emotion model built by fine-tuning MARBERT exhibited the best performance when evaluated on the AraNET$_{\mathrm{Emo}}$ dataset, achieving an F1-score of 75.18%.

The study by Abdelali et al. [55] illustrated the training of a Transformer model called QARiB, compared QARiB with three existing BERT models, and provided best practices for training language models. The paper presented various approaches for training five different QARiB models that varied in terms of preprocessing steps, the included Arabic varieties, and the size of the training data. The training data encompassed formal texts from Arabic newswire sources and informal texts from Twitter. Various preprocessing techniques were experimented with, including tokenization using Farasa, removal of diacritical marks and Kashida (word elongation), and normalization of different forms of Arabic letters. The QARiB PLMs were evaluated on various downstream tasks, including emotion recognition. The best-performing model achieved an F1-score of 46.80% when evaluated on the SemEval 2018 emotion classification dataset. The study revealed that the inclusion of both formal (MSA) and informal (diacritical Arabic) texts led to the best-performing models in downstream tasks. Additionally, the paper reported that word segmentation improved the model's performance on the named entity recognition task but not on emotion recognition.

The use of transfer learning for Arabic emotion recognition was investigated in a recent study by Althobaiti et al. [57]. The study also examined the incorporation of semantic information, including named entities and sentiments extracted from text, before fine-tuning the pretrained BERT-based model. Two methods were investigated: embedding the semantic information within the text and using an SEP token to differentiate the input text from the added semantic information. The proposed methods were thoroughly evaluated for various PLMs. The results indicated that including semantic information in the text before fine-tuning the model enhanced the performance of the PLM, regardless of the differences or similarities between the pretrained and fine-tuned data. The paper demonstrated that incorporating only named entity information yielded better results than incorporating only sentiment information. The proposed method was evaluated on the AETD dataset, resulting in an accuracy of 81.73%, a weighted-average F1-score of 81.60%, a precision of 81.92%, and a recall of 81.40%.

From the literature on Arabic emotion recognition, we can observe several notable issues related to fine-grained text classification in general and Arabic emotion recognition in particular. Fine-grained emotion recognition requires a large number of labeled data examples per emotion category to obtain good results, especially when working with deep learning models. Moreover, the degree of nuance required in fine-grained Arabic emotion recognition can vary from one application to another. For instance, some studies focused on six emotion categories, while others delved into 11 emotion categories in their investigations. In fact, one study [23] on English emotion recognition created labeled data for 27 emotion categories. These varying requirements from one study to another necessitate a significant amount of time and effort to collect and label more data or to modify existing datasets.

2.3. Arabic Emotion Recognition Datasets

The creation of labeled Arabic emotion recognition datasets has attracted a significant amount of attention in the literature. Abdul-Mageed and Korayem's study [58] introduced the multi-dialect DINA dataset for Arabic emotion analysis. This dataset, collected from Twitter, comprises 3000 tweets written in various dialects and is manually annotated according to Ekman's six emotion categories: anger, disgust, fear, happiness, sadness, and surprise. Another study by Al-Khatib and El-Beltagy [39] presented the AETD dataset, primarily consisting of Egyptian dialect tweets. AETD includes 10,065 examples, each labeled with one of seven emotion tags: anger, fear, happiness, love, sadness, surprise, or sympathy, or none in case of the absence of emotion in the given example. Additionally, Alhuzali et al. [59] introduced the LAMA dataset, which contains 7268 tweets covering Plutchik's eight basic emotions: anger, anticipation, disgust, fear, joy, sadness, surprise, and trust. The data collection process began with a seed set of 48 emotion-carrying expressions and regular expressions, resulting in the selection of 8000 tweets (1000 per emotion category) for manual review by annotators to remove duplicates, ultimately leading to the removal of 732 tweets from the final dataset.

One of the earliest labeled datasets for Arabic emotion recognition was created through the SemEval-2018 Task 1: Affect in Tweets challenge [42]. The subtask for Arabic multi-label emotion classification used a labeled dataset from Twitter, which consisted of 4381 tweets written in multiple Arabic dialects and covered eleven emotions: anger, anticipation, disgust, fear, happiness, love, optimism, pessimism, sadness, surprise, and trust. Additionally, the study by Almahdawi et al. [60] introduced the Iraqi Arabic Emotion Dataset (IAEDS), which comprised 1365 examples collected from Facebook posts and labeled with Ekman's six basic emotion categories.

Furthermore, the SenWave dataset, presented in the study by Yang et al. [61], includes 10,000 Arabic tweets annotated into ten categories. It was collected during the early months of the COVID-19 pandemic and encompasses a variety of Arabic dialects. The ArPanEmo dataset [62] is another Arabic emotion recognition dataset, consisting of 11,128 online posts, each labeled with one of eleven categories: anger, anticipation, confusion, disgust, fear, joy, neutral, optimism, pessimism, sadness, and surprise. It primarily focuses on the Saudi dialect and addresses topics related to the COVID-19 pandemic collected from three sources: Twitter, YouTube, and online newspaper comments between March 2020 and March 2022.

A comprehensive overview of Arabic emotion analysis, including available language models and emotion recognition resources, was presented in a recent study [22]. Table 1 provides a summary of existing datasets for Arabic emotion recognition that are manually or semi-automatically labeled.

Table 1. Summary of existing Arabic emotion recognition datasets.

	Details				
Dataset	**Annotation Method and Type**	**Source and Size**	**Topics**	**Emotion Categories**	**Arabic Variety**
DINA [58]	Semi-automatic (seed words), single-label annotation	Twitter (3000 tweets)	General	Ekman's 6 basic emotions: anger, disgust, fear, happiness, sadness, and surprise	Mixture of MSA and Arabic dialects
LAMA [59]	Semi-automatic (seed words), single-label annotation	Twitter (7268 tweets)	General	Plutchik's 8 basic emotions: anger, anticipation, disgust, fear, joy, sadness, surprise, and trust	Mixture of MSA and Arabic dialects
AETD [39]	Manual, single-label annotation	Twitter (10,065 tweets)	General	7 emotions: anger, sympathy, fear, joy, sadness, love, and surprise	Egyptian dialect
SemEval-2018 (subtask E-c) dataset [42]	Manual, multi-label annotation	Twitter (4381 tweets)	General	11 emotions: anger, anticipation, disgust, fear, happiness, love, optimism, pessimism, sadness, surprise, and trust	Mixture of MSA and Arabic dialects

Table 1. *Cont.*

Dataset	Details				
	Annotation Method and Type	**Source and Size**	**Topics**	**Emotion Categories**	**Arabic Variety**
IAEDS [60]	Manual, single-label annotation	Facebook (1365 posts)	General	Ekman's 6 basic emotions: anger, disgust, fear, happiness, sadness, and surprise	Iraqi dialect
SenWave [61]	Manual, multi-label annotation	Twitter (10K tweets)	COVID-19 pandemic	9 emotions: optimistic, thankful, empathetic, pessimistic, anxious, sad, annoyed, denial, and joking	Mixture of MSA and Arabic dialects
Hussien et al. [38]	Manual, single-label annotation	Twitter (2025 tweets)	General	4 emotions: joy, sadness, anger, and disgust	Mixture of MSA and Arabic dialects
Hussien et al. [38]	Semi-automatic (emojis, hashtags, and lexicon), single-label annotation	Twitter (22,752 tweets)	General	4 emotions: joy, sadness, anger, and disgust	Mixture of MSA and Arabic dialects
AraNET$_{\text{Emo}}$ dataset [63]	Manual and semi-automatic (seed phrases expressing emotions), single-label annotation	Twitter (191K tweets)	General	Plutchik's 8 basic emotions: anger, anticipation, disgust, fear, joy, sadness, surprise, and trust	Mixture of MSA and Arabic dialects
ArPanEmo dataset [62]	Manual, single-label annotation	Twitter, YouTube, and online newspaper comments (11,128 online posts)	COVID-19 pandemic (health care, impacted life aspects)	10 emotions: anger, anticipation, confusion, disgust, fear, joy, optimism, pessimism, sadness, and surprise	Saudi dialect

3. Methodology

Our approach, which aimed at enhancing the performance of low-resource, fine-grained text classification, primarily entails two components: a stacking ensemble of diverse models and self-training, as illustrated in the general framework shown in Figure 1.

The proposed approach does not require a considerable amount of labeled training data. A small quantity of labeled data for each emotion category proves sufficient to initiate the entire process. As the two components of the proposed method undergo iterations, the performance of the trained classifier (i.e., the stacking ensemble model) improves, along with an increase in the volume of labeled data automatically generated throughout the entire process.

The first component is a meta-ensemble technique that uses a two-level architecture to conduct emotion recognition. The first level aims to create a set of diverse models using different learning paradigms, including conventional machine learning models, few-shot models, zero-shot classification models, and transfer learning models. This diversity can result in more accurate and comprehensive predictions; that is, each type of learning model may excel in different types of data and tasks [64]. The second level involves constructing a higher-level model (meta-learner or stacker) that learns how to effectively combine the predictions of the base models to improve the overall performance on Arabic emotion recognition.

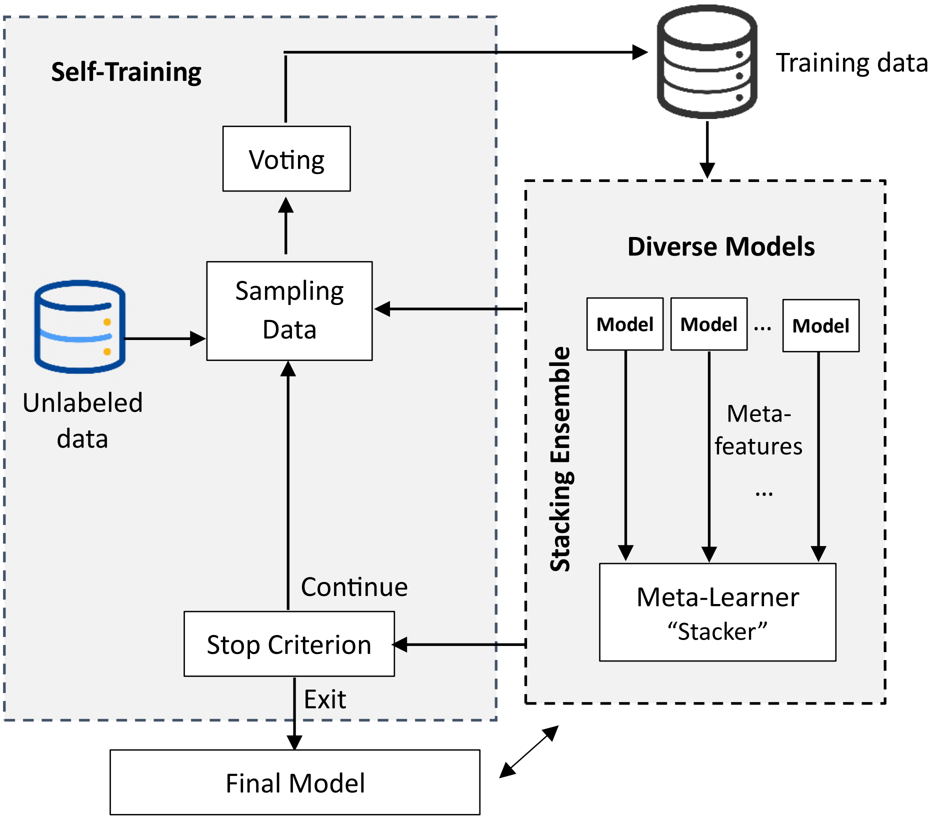

Figure 1. General framework we proposed to boost low-resource, fine-grained emotion recognition.

The second component of our proposed framework is responsible for the iterative aspect of the method. Self-training involves using the base classifiers built in the first component to iteratively label unlabeled data and then incorporating the confident predictions into the training dataset. To determine the most confident predictions, the algorithm applies majority voting to select the predictions with the highest number of votes. As iterations continue, the base classifiers collaborate to improve their performance, and consequently, the meta-learner also continues to improve. The stopping criterion can be set as either a specific number of iterations or when performance plateaus, indicating that the meta-learner's performance has reached a plateau on a validation or development set.

More details about the architecture of our proposed method are illustrated in Figure 2. The following sections explain in detail each component of the proposed method.

It is worth noting that our proposed method is not specifically designed to work only for emotion recognition tasks or for the Arabic language; rather, it is applicable to any language and various fine-grained text classification tasks. However, the method relies on the availability, diversity, and richness of unlabeled data that can be confidently labeled during self-training; that is, if the dataset does not effectively represent the desired classes, the method's effectiveness could be restricted. Additionally, some models in our proposed method, such as the zero-shot classification model, few-shot methods, and transfer learning model, depend on the similarity between the source and target domains [56]. Therefore, if the source data utilized in pretraining the models is dissimilar to the target domain of the given task, our proposed method may not yield optimal results, limiting its applicability.

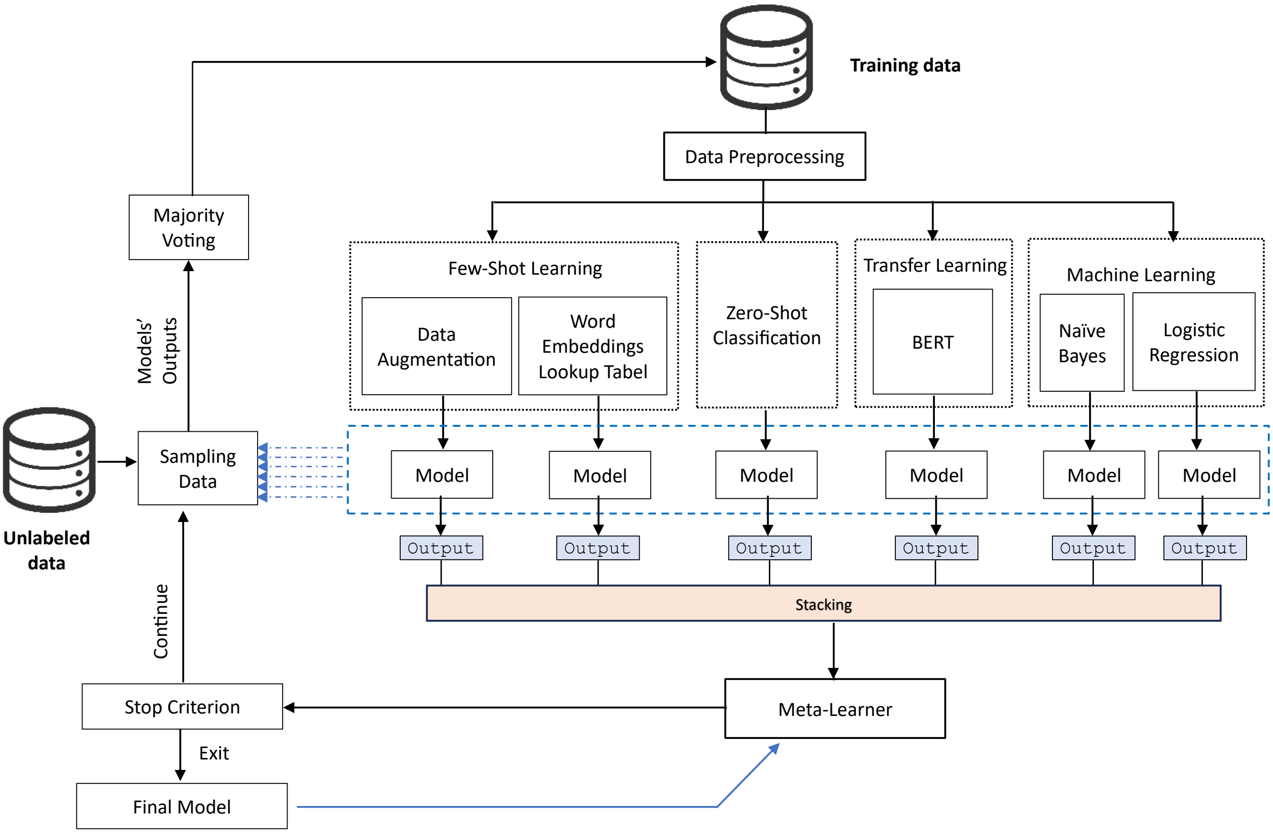

Figure 2. The architecture of the proposed method.

4. Diverse Model Stacking Ensemble

Stacking, also known as stacked generalization, is an ensemble learning technique that involves creating multiple base models, often using different learning algorithms, and then combining their predictions using a higher-level model called the meta-learner or stacker [64–66]. Each base classifier is trained on the same training dataset. The meta-learner is trained on a separate dataset created using K-fold cross-validation; that is, for each fold, the base models are trained on a subset of the dataset and then utilized to make predictions on the remaining dataset. Finally, the base models' predictions for each fold are combined into a new dataset, also know as meta-features, to be used for training the meta-learner [67]. The goal of the stacking ensemble model is to exploit the diversity and complementary strengths of the base classifiers to improve overall predictive performance [68].

We created six base models in our proposed method, as shown in Figure 2. The models were created based on machine learning, transfer learning via BERT, few-shot methods, and zero-shot classification. The diverse set of learning paradigms may help create models that are able to capture different patterns from the data and consequently have different strengths and characteristics. The traditional machine learning algorithms can perform reasonably well with a smaller amount of labeled data compared to complex deep learning models, such as Transformer-based models [69]. On the other hand, transfer learning models, such as BERT-based models, capture deep semantic relationships, making them highly effective at understanding complex context and nuances in text. In addition, unlike traditional machine learning, fine-tuned BERT models can generalize well to new or unseen data even if the distribution shifts significantly [70].

Few-shot methods, including techniques such as nearest neighbor embedding lookup and data augmentation, are powerful solutions for mitigating the challenges posed by small amounts of labeled data. Data augmentation increases data diversity by creating new training examples through the application of various transformations on the existing labeled data. This process enables the model to generalize better to unseen textual examples with new vocabularies. At the same time, augmentation reduces the risk of overfitting on the limited labeled data by introducing variability into the training data [71,72]. On the other hand, nearest neighbor embedding lookup leverages the idea that similar examples in the feature space should have similar labels. Leveraging the inherent structure of existing data can lead to improved generalization and performance [73]. Zero-shot learning allows the model to extend its capabilities to new classes without additional labeled data or retraining. It encourages the model to learn high-level semantic relationships between classes and to recognize patterns and attributes that are common across different classes [74].

To build a meta-learner for combining the predictions of the base classifiers, we employed SVM. The following sections illustrate the details of each base classifier utilized in our proposed method.

4.1. Few-Shot Learning

We utilized two methods of few-shot learning: data augmentation and embedding lookups in order to build few-shot models for Arabic emotion recognition in our proposed method, as illustrated in Figure 2.

4.1.1. Data Augmentation

One simple way to enhance the performance of Arabic emotion recognition on small labeled data is to apply data augmentation techniques to generate new labeled training examples from the existing examples [71,72]. Indeed, data augmentation techniques are utilized to artificially expand the variety and quantity of a training dataset by performing various transformations or alterations to the original data. In the context of text data, the two most frequent forms of data augmentation techniques are back translation and token perturbations [75]. Back translation entails the machine translation of text from the source language into one or more target languages. The translated text is then translated back into the original source language. This process can introduce variations in phrasing and word usage [76]. Token perturbations involve performing simple transformations, such as random synonym replacement, word swap, insertion, or deletion [72]. Tables 2 and 3 show examples of different types of data augmentation techniques when applied to MSA and dialectal Arabic texts, respectively.

Table 2. Examples of different data augmentation techniques when applied to MSA text.

Augmentation	**Sentence**
None	أعلنت غوغل عن هذه الإجراءات في فبراير وسيتم إطلاقها عالميا في أغسطس الجاري English gloss: Google announced these procedures in February and will launch them globally this August
Synonym replacement	أعلنت غوغل توفر هذه الإجراءات 17 فبراير وسيتم تحقيقها عالميا في أغسطس الجاري English gloss: Google announced the availability of these procedures on February 17, and they will be achieved globally this August
Word insertion (Random)	أعلنت غوغل عن كامل هذه الإجراءات في فبراير وسيتم إطلاقها عالميا في أغسطس الجاري English gloss: Google announced all of these procedures in February and will launch them globally this August
Word swap (Random)	أعلنت عن غوغل هذه الإجراءات في فبراير وسيتم عالميا إطلاقها في الجاري أغسطس English gloss: Google announced these procedures in February, and they will be launched globally this August
Word deletion (Random)	أعلنت غوغل عن في فبراير وسيتم عالميا في أغسطس الجاري English gloss: Google announced in February and will be globally this August
Back Translation (English)	أعلن جوجل عن هذه الإجراءات في شباط/فبراير وسيتم إطلاقها عالمياً في آب/أغسطس English gloss: Google announced these procedures in February and will launch them globally in August

We can generate a number of new examples with data augmentation techniques. Following the aforementioned five data augmentation techniques, the dataset examples will multiply six times. Then, a machine learning algorithm (e.g., Naive Bayes) can be trained on the expanded training set with the examples that resulted from the data augmentation techniques.

Table 3. Examples of different data augmentation techniques when applied to dialectal Arabic text.

Augmentation	Sentence
None	الامثال هى خزان لتجارب الشعوب و لحكمتها اللى بتلخصها ف كلام موجز عشان تنقلها من جيل للتانى English gloss: Proverbs are a reservoir of people's experiences and wisdom, which are summed up in brief words in order to pass them on from one generation to the next.
Synonym replacement	الامثال هى خزان لتجارب التاريخ و لحكمتها هي بتلخصها فيه شيئ بسيط عشان تستخدمها لأى جيل للتانى English gloss: Proverbs are a reservoir of history's experiences, and for its wisdom, it sums it up in something simple, so that it can be used by any generation to the next
Word insertion (Random)	الامثال هى خزان لتجارب الشعوب و لحكمتها الإنسانية اللى بتلخصها ف كلام موجز عشان تنقلها من جيل للتانى English gloss: Proverbs are a reservoir of peoples' experiences and their human wisdom, which are summed up in brief words in order to pass them on from one generation to the next.
Word swap (Random)	الامثال هى خزان لتجارب الشعوب و اللى بتلخصها لحكمتها ف كلام موجز تنقلها من عشان للتانى جيل English gloss: Proverbs are a reservoir of people's experiences, which summarize their wisdom in brief words that you pass on from one generation to the next
Word deletion (Random)	هى خزان لتجارب الشعوب و لحكمتها اللى بتلخصها ف تنقلها من جيل English gloss: It is a reservoir of peoples' experiences and wisdom that summarizes them and transmits them from generation to generation
Back Translation (English)	والمثل العليا هي مستودع لتجارب الشعوب وحكمتها، التي هي، بكلمات مختصرة، وهي تنتقل من جيل إلى جيل. English gloss: Ideals are a repository of peoples' experiences and wisdom, which, in short, are passed down from generation to generation.

4.1.2. Embedding Lookup

Large language models have demonstrated their worth in addressing natural language processing tasks when there is a scarcity of labeled data [73]. The reason is that these models learn useful text representations (embeddings) that capture semantic and contextual information about the words. Mapping discrete tokens, such as words or characters, to continuous vector representations in a lower-dimensional space is called an embedding lookup [14,15,77].

The embeddings of large language models can be exploited to classify text based on searching for nearest neighbors among the embeddings. This technique exploits the notion that words or tokens with similar meanings or contexts should have similar vector representations in the embedding space. There are three phases in this procedure:

- Employing the language model to create embeddings for labeled texts.
- Performing a search for the nearest neighbors among the stored embeddings.
- Aggregating the labels of the nearest neighbors to obtain a prediction.

The primary benefit of employing the nearest neighbor embedding lookup technique is that there is no need to train or fine-tune a model to harness the few existing labeled data [73,78,79]. An illustration of how one can perform nearest neighbor embedding lookup for emotion recognition is shown in Figure 3.

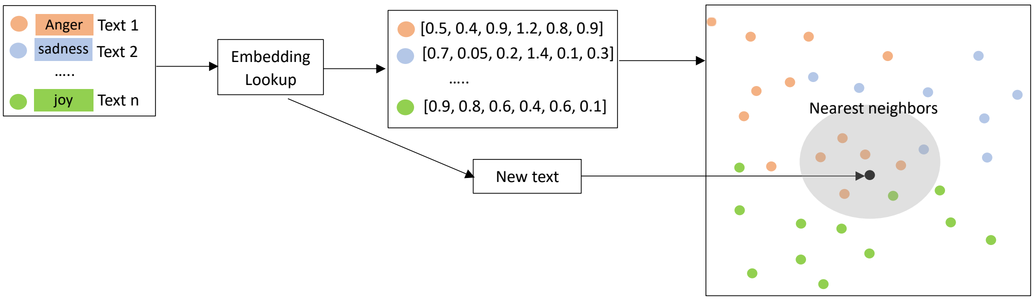

Figure 3. An illustration of nearest neighbor embedding for emotion recognition.

It is essential to provide the proper number of neighbors to search for and retrieve in this approach since choosing too few neighbors might bring noise, while choosing too many neighbors could cause blending with nearby clusters [15]. After some pilot experiments, we opted to retrieve 15 text examples as neighbors and subsequently selected the emotion prediction that appeared at least 5 times. These parameter choices are particularly well-suited for the nuanced nature of fine-grained emotion recognition.

4.2. Zero-Shot Classification

Zeros-shot classification allows a model to classify texts into a predefined set of classes, even if those classes were not seen during the model's training process. It enables a model to generalize its understanding of text and make predictions for classes it has never encountered before [74]. The goal of zero-shot text classification is to use a pretrained model without further fine-tuning on any task-specific dataset. One strategy to accomplish this objective is to employ carefully crafted prompts alongside a pretrained model that has expertise with a task more closely related to the one being addressed [79].

Zeros-shot emotion recognition can be addressed by adapting a model that has been fine-tuned on a task close to text classification, such as text entailment [79]. In text entailment, the model needs to determine whether two text passages are likely to follow or contradict each other [80]. Indeed, the models are typically trained to detect entailments and contradictions with natural language inference (NLI) datasets such as the Multi-Genre NLI (MNLI) dataset [81] or the Cross-Lingual NLI (XNLI) dataset [80]. Each example in these datasets is composed of three elements: a premise, a hypothesis, and a label. The label can be one of three classes: *entailment*, *neutral*, and *contradiction*. The entailment label is used when the hypothesis text is true under the premise. The contradiction label is assigned when the hypothesis is false or inappropriate under the premise. The neutral label is used if neither of these cases is applicable. Tables 4 and 5 present examples of text entailment with three classes in English and Arabic, taken from MNLI and XNLI, respectively.

Table 4. Examples of three classes of text entailment for English taken from MNLI dataset.

Premise	Hypothesis	Label
but that takes too much planning	It doesn't take much planning	contradiction
we have gone on trips we've bathed in streams	In addition to bathing in streams, we've also gone to spas and saunas	neutral
how can you prove it	Can you tell me how to prove it?	entailment

Table 5. Examples of the three classes of text entailment for Arabic taken from XNLI dataset.

Premise	Hypothesis	Label
مجرد حدس English gloss: Simply intuition	انه مجرد تخمين English gloss: It is just a guess	entailment
ولكن هذا يتطلب الكثير من التخطيط English gloss: But this requires a lot of planning	انها لا تاخذ الكثير من التخطيط English gloss: It does not take a lot of planning	contradiction
واتفق المشاركون عموما على ان التحسينات في ادارة الشركات ستجلب تحسينات في مراجعة الحسابات English gloss: participants generally agreed that improvements in corporate management will lead to enhancements in the audit process.	واتفق المشاركون فقط على عدد قليل من المواضيع English gloss: Participants agreed only on a few topics	neutral

The key idea in order to build an emotion recognizer using a model trained on the MNLI or XNLI datasets without requiring any labeled data is to treat the text to be classified as the premise and then formulate the hypothesis template as:

This text is {label}.

Instead of *label* in the default hypothesis template mentioned above, one can insert the emotion class name. The entailment score is used to determine how likely that premise is to be about that class. This process can be run for any number of classes sequentially. For Arabic emotion recognition, we translated the aforementioned hypothesis template into Arabic.

4.3. Transfer Learning

Transfer learning in NLP involves leveraging linguistic knowledge acquired from a source text domain or task to improve the performance of a target text domain or task.

Architecturally, this entails dividing the model into a body and a head, with the head being a task-specific network. The weights of the body learn significant features of the source domain during training, and these weights are used to initialize a new model for the new task [79,82].

The foundation of transfer learning in NLP is the notion of pretrained language models. These models, typically based on deep neural networks, such as the Transformer architecture [77], develop detailed contextualized representations of words and phrases from enormous text corpora. The gained knowledge is stored in model parameters, making it possible to transfer this knowledge to various downstream tasks [79,82]. Adjusting the pretrained model's parameters using labeled data from the downstream task domain is called fine-tuning [82]. The fine-tuning process enables the model to specialize on details of the target task while preserving the overall language understanding abilities learned from the source domain [82]. Both GPT [15] and BERT [14] are prominent examples of pretrained language models that have been extensively utilized and proven efficient for transfer learning in various NLP tasks [57,83–85].

4.4. Machine Learning

4.4.1. Feature Extraction

For conventional ML models, the Term Frequency–Inverse Document Frequency (TF-IDF) with word unigrams is used for text extraction and representation.

Word unigrams are individual words where each word is treated as a separate feature. Word unigrams are chosen to mitigate issues arising from high dimensionality and sparsity, factors that are particularly relevant to our early-stage approach where labeled data are typically limited.

The TF-IDF is one of the most commonly used approaches for text feature extraction. It is a statistical measure used to calculates the importance of a word within a text example of a specific emotion class relative to its frequency in the entire corpus across all emotions [86]. The Term Frequency (TF) counts the number of times a word occurs in text examples belonging to a particular emotion class, while the Inverse Document Frequency (IDF) diminishes words that appear very frequently in the entire corpus across all emotion classes and increases the weight of words that appear rarely [87]. The mathematical formula for TF-IDF is as follows:

$$TF - IDF(w, class, Corpus) = TF(w, class) * IDF(w, Corpus) \tag{1}$$

where w indicates the word. The complete mathematical formula for IDF is as follows:

$$IDF(w, Corpus) = log \frac{N}{1 + n(w, Corpus)} \tag{2}$$

where N refers to the number of examples in the corpus and $n(w, Corpus)$ is the number of examples in the corpus that contain the word w.

4.4.2. Naive Bayes

Naive Bayes is a conditional probability model based on the Bayes theorem. The "Naive" part of its name refers to the assumption of independence among features. It assumes that the features are conditionally independent, which means that the presence or absence of one feature does not impact the presence or absence of another. This assumption simplifies the calculations and makes the algorithm computationally efficient [88,89].

Naive Bayes assigns probabilities $P(C_k \mid x_1, \ldots, x_n)$ for each of the K possible classes C_k. It computes the probability of a given example with features $x = (x_1, \ldots, x_n)$ belonging to a particular class k based on the probabilities of features given the class label as follows:

$$P(C_k|x) = P(x|C_k) * P(C_k)/P(x) \tag{3}$$

where $P(C_k|x)$ is the probability of class C_k given the feature x, $P(x|C_k)$ is the probability of feature x given class C_k, $P(C_k)$ is the prior probability of class C_k, and $P(x)$ is the probability of feature x.

Naive Bayes is commonly utilized for text classification [90,91]. The Naive Bayes classifier combines the Naive Bayes probability model with a decision rule, often utilizing the Maximum A Posteriori (MAP) decision rule as a standard approach. Consequently, the Bayes classifier becomes a function that designates a class label $\hat{y} = C_k$ to a data point x by maximizing

$$\hat{y} = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}} \; P(C_k) \prod_{i=1}^{n} P(x_i \mid C_k) \tag{4}$$

4.4.3. Logistic Regression

Logistic regression is a statistical model utilized for binary classification, which involves predicting the probability that an example belongs to a particular class. The model uses the sigmoid function (also known as the logistic function) to map the output of a linear equation into a probability value between 0 and 1. The sigmoid function has an S-shaped curve and is defined as

$$\sigma(z) = \frac{e^z}{e^z + 1} = \frac{1}{1 + e^{-z}} \tag{5}$$

where z is the linear combination of the input features and their respective weights.

The logistic regression hypothesis function models the probability that an example belongs to a particular class (e.g., 1) as follows:

$$P(y = 1|x) = \sigma(\theta^T x) \tag{6}$$

where θ represents the parameter vector (weights) and x represents the input features.

The logistic regression model is trained by minimizing a cost function, which quantifies the error between the predicted probabilities and actual labels. The most commonly used cost function is the log-loss (also known as cross-entropy loss) for binary classification:

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} (y^{(i)} log(h_\theta(x^{(i)})) + (1 - y^{(i)}) log(1 - h_\theta(x^{(i)}))) \tag{7}$$

where m is the number of training examples, $h_\theta(x^{(i)})$ is the predicted probability, and $y^{(i)}$ is the true class label for the i-th example. To minimize the cost function, gradient descent or other optimization methods are employed to update the parameter vector θ iteratively [92,93].

Logistic regression is a binary classification algorithm, and in order to use it for fine-grained emotion recognition, we use the One-vs.-Rest (OvR) approach, also known as the One-vs.-All approach. In this approach, we create multiple binary classifiers, one for each emotion class. During prediction, the algorithm runs the input through all classifiers and selects the class for which the classifiers give the highest probability of being in that class [94,95].

5. Self-Training

One of the semi-supervised learning techniques is self-training, which is utilized to improve the performance of a model by having it iteratively learn from its own predictions. In this technique, a model starts with a small amount of labeled data and gradually increases its size by pseudo-labeling unlabeled data using its own predictions. Over these iterations, the model's performance and generalization capabilities can improve as it incorporates more data. Additionally, the size of the labeled training data will be multiplied by the end of the whole process of the iterations [96,97]. Therefore, the self-training process typically involves the following steps:

1. Initialization: A model is trained on limited labeled data.
2. Pseudo-labeling: The trained model is then used to predict the labels of unlabeled data. These predictions are treated as pseudo-labels.
3. Expanded dataset: The pseudo-labeled data are combined with the original labeled dataset to create a new larger training dataset.
4. Retrained model: The model is retrained on the expanded dataset. This process helps the model to learn from its own predictions.
5. Iteration: Steps 2 to 4 are repeated for a predefined number of iterations or when performance plateaus.

We propose to use self-training in our method, as illustrated in Figures 1 and 2. Instead of relying on the predictions of a single model, self-training in our proposed method employs the majority voting of the six base models to determine the final label for each example. Initially, each of the six base models predicts labels for unlabeled data. Each base model's prediction is treated as a vote, and the label with the most votes becomes the pseudo-label. To ensure the reliability of pseudo-labels, we set a confidence threshold of 5. This means that only predictions that receive 5 agreements (i.e., exceed the threshold) among the models are considered confident enough to be included to the training set for the next iteration.

At each iteration, not only are the base models retrained; the SVM meta-learner, which takes advantage of the new predictions from the base models, is also retrained. In our proposed method, the process of pseudo-labeling, majority voting, and retraining is iteratively repeated for a predefined number of iterations. The final model is the SVM meta-learner in the last iteration.

6. Experiments

In this section, we first present the utilized data, evaluation metrics, and experimental setting. Then, we conduct experiments to answer the following questions:

RQ1: *How does the performance of our proposed method compare to that of the state-of-the-art methods in the literature?*

RQ2: *What are the influences of the two primary components of our proposed method, namely the diverse model stacking ensemble and self-training?*

RQ3: *How does the performance of our proposed method compare to that of each base model trained using different learning paradigms in low-resource scenarios?*

6.1. Datasets

This section presents the datasets utilized to evaluate the performance of our proposed method for Arabic emotion recognition. We utilized two labeled datasets: (a) the ArPanEmo dataset [62] and (b) AETD [39]. More details about the ArPanEmo and AETD datasets can be found in Section 2.3. Additionally, we employed a large amount of unlabeled data for the self-training part of our proposed method. We utilized the Arabic Sentiment Tweets Dataset (ASTD) [98] after removing its labels. The ASTD dataset, collected from Twitter, is divided into four categories: objective, subjective negative, subjective positive, and subjective mixed. We removed the sentiment analysis labels and used only the text examples, comprising 10,006 tweets. Additionally, we randomly collected 50,000 tweets from Twitter using the Python Tweepy library [62] to utilize the Twitter API. The tweets were collected in separate periods over the past two years. The total amount of unlabeled data was 60,006 examples.

6.2. Data Preprocessing

The data were preprocessed before conducting experiments and building models in order to remove unnecessary characters from the raw text and normalize letters that are usually written interchangeably, which can lead to data sparsity. In addition, the utilized data were derived from online content, which made it full of noise, such as URLs and symbols. The preprocessing steps we applied include:

- Eliminating unwanted characters and symbols, such as mentions, URLs, hashtags, and retweets.
- Reducing letter repetition within words.
- Removing non-Arabic words.
- Omitting diacritical marks and punctuation.
- Removing Arabic letter elongations.
- Substituting underscores in hashtag texts with spaces.
- Normalizing different forms of *Alif* to its base form, a bare *Alif*; *Alif maqSwrah* to dotted *yA'*; and *tA' marbutah* to *hA'*.

We utilized regular expressions to reduce letter repetition within words, URLs, mentions, retweets, and hashtag symbols. Regarding the removal of diacritical marks and punctuation, as well as the normalization of the different forms of letters, we used the normalizer provided by the AraNLP library [99].

6.3. Evaluation Metrics

The standard metrics widely used for evaluating the performance of classification models include *precision* (*P*), *recall* (*R*), *F1-score* (*F1*), and *accuracy*. Given the built emotion recognition model and the set of emotion classes M, we can compute the evaluation metrics to evaluate the model's performance with respect to each emotion label m as outlined below:

$$Accuracy_m = \frac{TP_m + TN_m}{TP_m + TN_m + FP_m + FN_m}$$

$$Precision_m = P_m = \frac{TP_m}{TP_m + FP_m}$$

$$Recall_m = R_m = \frac{TP_m}{TP_m + FN_m}$$

where TP_m represents the true positives, signifying the number of examples in the dataset correctly identified by the model as emotion m, while TN_m represents the true negatives, denoting the number of examples correctly identified by the model as *not* belonging to emotion m. On the other hand, FP_m denotes the false positives, indicating the number of examples incorrectly labeled by the model as emotion m, and FN_m represents the false negatives, signifying the number of examples erroneously labeled by the model as *not* belonging to emotion m.

The F1-score is a balanced performance metric that calculates the harmonic mean of precision and recall. The equation of the F1-score is:

$$F1\text{-}score_m = F1_m = 2 * \frac{P_m * R_m}{P_m + R_m}$$

To evaluate the overall performance of the models we developed in our experiments and to compare their performance with state-of-the-art methods from the literature, we must assess the model's performance across multiple metrics. These metrics include accuracy, as well as the macro average, micro average, and weighted average of three key metrics: precision, recall, and F1-score. This is conducted as follows:

$$Accuracy = \frac{\sum_{m=1}^{M} TP_m + \sum_{m=1}^{M} TN_m}{\sum_{m=1}^{M} TP_m + \sum_{m=1}^{M} TN_m + \sum_{m=1}^{M} FP_+ \sum_{m=1}^{M} FN_m}$$

The macro average for the *precision*, *recall*, and *F1-score* metrics are determined by computing the average of these metrics for each emotion class separately, assigning equal importance to each class. This method treats every emotion class equally, regardless of its frequency, and the calculations are carried out using the following formulas:

$$P_{macro} = \frac{\sum_{m=1}^{M} P_m}{|M|}$$

$$R_{macro} = \frac{\sum_{m=1}^{M} R_m}{|M|}$$

$$F1_{macro} = 2 * \frac{P_{macro} * R_{macro}}{P_{macro} + R_{macro}}$$

where $|M|$ denotes the total number of emotion classes present in the dataset.

The micro average focuses on the overall performance across all classes, with a significant emphasis on dominant classes, assigning them greater weight. The calculations for the micro average are determined by the following formulas:

$$P_{micro} = \frac{\sum_{m=1}^{C} TP_m}{\sum_{m=1}^{M} TP_m + \sum_{m=1}^{M} FP_m}$$

$$R_{micro} = \frac{\sum_{m=1}^{M} TP_m}{\sum_{m=1}^{M} TP_m + \sum_{m=1}^{M} FN_m}$$

$$F1_{micro} = 2 * \frac{P_{micro} * R_{micro}}{P_{micro} + R_{micro}}$$

The weighted average considers the proportion of each emotion class when computing the metric averages. It strikes a balance between micro and macro averaging, taking into account both the overall performance and the significance of each class. These calculations are determined using the following formulas:

$$P_{weighted} = \frac{\sum_{m=1}^{M} P_m * |m|}{\sum_{m=1}^{M} |m|}$$

$$R_{weighted} = \frac{\sum_{m=1}^{M} R_m * |m|}{\sum_{m=1}^{M} |m|}$$

$$F1_{weighted} = \frac{\sum_{m=1}^{M} F1_m * |m|}{\sum_{m=1}^{M} |m|}$$

where $|m|$ represents the total number of examples in the dataset belonging to the emotion class m and $\sum_{m=1}^{M} |m|$ denotes the summation of the total examples for each emotion class in the dataset, which is equal to the dataset's overall size.

6.4. Experimental Setup

Regarding our experiments involving self-training, we set the number of iterations to 20, and in each iteration, a sample of 3000 unlabeled data was automatically annotated.

We evaluated the efficiency of our proposed method in the context of Arabic emotion recognition; therefore, we selected pretrained language models specifically tailored for the Arabic language. The chosen models were pretrained on a substantial corpus of Arabic text, encompassing both MSA and dialectal Arabic.

For few-shot models, particularly in one of the methods utilized in our study, the nearest neighbor embedding lookup method, we selected the MARBERTv2 large model [56] as a pretrained model. This model was employed to generate a single embedding vector for each text in the training dataset. In order to perform similarity searches of the embeddings, we used the FAISS index, an index created and managed using the Facebook AI Similarity Search (FAISS) library [100], which presents efficient algorithms to quickly search and cluster embedding vectors [78]. In the data augmentation technique, sentence transformations such as word swaps and deletions were randomly carried out using the *NLPAug* Python library. Additionally, contextual word embeddings of a language model were leveraged to introduce modifications to sentences, including synonym replacement and word insertion. The AraBERTv0.2-Twitter model [51] was utilized for this purpose. We intentionally opted for a different model than the one used in the aforementioned embedding lookup method, with the aim of increasing the variety among the models used. For back translation, neural machine translation models were used to translate between Arabic (ar) and English (en) in both directions. Specifically, the *OPUS-MT-tc-big-ar-en* and *OPUS-MT-tc-big-en-ar* models were utilized for these translations [101,102].

For zero-shot classification, we utilized the XLM-RoBERTa-large-xnli model as a text entailment model. This model was built by fine-tuning the XLM-RoBERTa-large model [103] on a combination of XNLI data in 15 languages, including Arabic. The XLM-RoBERTa-large model itself is a multilingual language model pretrained on text in 100 languages.

To apply transfer learning and construct a fine-tuned model for our proposed approach, depicted in Figure 2, we conducted fine-tuning of the AraBERTv0.2-Twitter model [51] on the training set in our experiments.

We utilized scikit-learn to construct machine learning models, specifically SVM, Naive Bayes, and Logistic Regression. In the case of multinomial Naive Bayes, we employed an additive smoothing parameter (alpha) set to 1. For Logistic Regression, we set the regularization parameter (*C*) to 0.01. As for SVM, we configured the regularization parameter (C) to 1, the kernel to 'rbf' (radial basis function), and the gamma to 'scale'.

We determined the training and test sets from each labeled dataset as follows:

Training: Our proposed method does not require a large amount of labeled data to initiate the algorithm, as illustrated in Figure 2. It only necessitates a small number of labeled examples to initiate the training process. Consequently, we selected only 10 examples per emotion class from each dataset as a starting point. Therefore, for ArPanEmo (which has 11 classes), we initiated with 110 labeled examples, while for AETD (with 8 classes), we initiated the training process of our proposed method with 80 labeled examples. Bear in mind that during iterations, more training examples are added from unlabeled data after being auto-labeled using the six base models and majority voting, as explained in Section 6.1.

To train the meta-learner in our proposed method, we applied a 2-fold cross-validation, where the six base models were trained on one fold and the second fold was used to generate predictions. The predictions made by the six base classifiers on the entire training dataset were employed as input features to train the meta-learner, as explained in Section 4.

Test: To evaluate our proposed method on the ArPanEmo dataset, we utilized a holdout test set. The set of ArPanEmo for testing was originally specified and released with the dataset by [62], allocating 20% of the dataset for the test set, which contains 2227 tweets. Regarding the AETD dataset, the exact set for testing is not specified by [39]. Therefore, we employed a 5-fold cross-validation approach for evaluating the data (not for training). This involved removing the 80 examples used in the training data, as mentioned earlier, and dividing the remaining AETD dataset into 5 folds. We then evaluated our proposed method on each fold and computed the average evaluation metrics. This allowed us to compare the performance of our proposed method with methods in the literature that utilized ArPanEmo and AETD for evaluation.

The PC used for conducting the experiments has a GPU (Nvidia GeForce RTX 3070), CPU @ 4.01 GHz, RAM with 32.0 GB, and a 952 GB SSD hard disk.

6.5. State-of-the-Art Performance Comparison (RQ1)

In this section, we compare our proposed method with other proposed methods in the literature for Arabic emotion recognition using benchmark datasets, namely the AETD and ArPanEmo datasets. Table 6 shows the comparison between previous studies on Arabic emotion recognition and our proposed approach on the AETD dataset.

Table 6. Comparisons between previous studies on Arabic emotion recognition and our proposed approach when tested on the AETD dataset.

Model	Accuracy	Weighted		
		F1	P	R
HF + DF [47]	71.8	71.8	72.2	71.8
Complement Naïve Bayes [39]	68.12	65.8	68.8	68.1
Sequential Minimal Optimization [39]	63.43	63.7	64.3	63.4
BERT [57]	78.93	78.93	78.97	78.97
BERT Integrated with Named Entity Recognition and Sentiment Analysis [57]	81.73	81.6	81.92	81.4
Our Proposed Approach	**83.15**	**83.19**	**83.31**	**83.15**

As seen in the results presented in Table 6, the study by [47], which proposed a hybrid model for Arabic emotion recognition, achieved a weighted average F1-score of 71.8%. Their method combines both human-engineered feature-based and deep feature-based models, as detailed in Section 2. A recent study conducted by [57] showed that employing transfer learning to simply fine-tune a BERT-based model for emotion recognition using the AETD dataset resulted in a weighted average F1-score of 78.93%. Moreover, incorporating semantic information, such as Named Entity Recognition and sentiment analysis, into the BERT-based model during the fine-tuning process yielded a higher weighted average F1-score of 81.60%. Our proposed method outperformed these aforementioned methods when tested on the AETD dataset, achieving a weighted average F1-score equal to 83.19%.

For the ArPanEmo dataset, Table 7 shows the comparisons between our proposed method and other previous studies when tested on the ArPanEmo dataset.

Table 7. Comparisons between previous studies on Arabic emotion recognition and our proposed approach when tested on the ArPanEmo dataset.

Model	Accuracy	Weighted			Macro		
		F1	P	R	F1	P	R
BERT [57]	67	67	69	67	66	68	67
BiGRU [57]	40	40	48	39	38	44	39
SVM [57]	58	58	68	58	55	69	52
Our Proposed Approach	**72.28**	**72.12**	**72.81**	**72.28**	**70.63**	**72.47**	**69.83**

The ArPanEmo dataset was examined in the literature [62] for Arabic emotion recognition using SVM, a deep learning method (BiGRU), and transfer learning through fine-tuning a pretrained model (BERT), achieving weighted average F1-scores of 58%, 40%, and 67%, respectively. Our proposed method outperformed the previously mentioned methods, achieving the highest weighted average F1-score of 72.12%, despite starting the training process with only 10 examples per emotion class in addition to automatically annotated examples from unlabeled data during training iterations.

6.6. Ablation Experiments (RQ2)

Our proposed method depends on two main components: (a) a stacking ensemble model that combines the predictions from diverse models employing various learning paradigms, and (b) self-training using majority voting among the diverse models to automatically expand the set of labeled training data, as shown in Figure 1. In order to dissect the impact of these two components and provide a deeper understanding of their individual contributions to the overall results, we conducted ablation experiments.

In the first experiment, we examined the influence of the stacking ensemble model. We utilized the complete manually labeled training data of the benchmark datasets (i.e., a high-resource scenario). In the second experiment, we conducted an analysis of the performance of self-training with majority voting, isolating it from the stacking ensemble model. During the self-training experiments, we did not use the entire set of training labeled data and, instead, initiated the training process with only 10 labeled examples per emotion class from the benchmark datasets.

In both ablation experiments, we developed six base classifiers, which included Naive Bayes, Logistic Regression, few-shot methods (i.e., nearest neighbor embedding lookup and data augmentation), a zero-shot classification model, and a transfer learning-based model using BERT. These diverse base classifiers are necessary for building both the stacking ensemble model and self-training with majority voting. Consequently, we also examined their performances in both settings (i.e., with stacking ensemble in a high-resource condition and with self-training using majority voting). The following subsections illustrate the results of the ablation study.

6.6.1. Analysis of our Diverse Model Stacking Ensemble in a High-Resource Scenario

We developed a stacking ensemble of the aforementioned six base classifiers. All of these models were trained using the complete manually labeled training data of the benchmark datasets. The performance of the stacking ensemble model and that of the individual base classifiers on the AETD and ArPanEmo datasets is presented in Tables 8 and 9, respectively.

Table 8. Performance of the stacking ensemble model and base classifiers on the AETD dataset.

Model	Accuracy	Weighted			Training Data (Size)
		F1	P	R	
Zero-Shot Classification	32.08	33.60	38.45	34.92	No-training
Naive Bayes	66.95	65.41	61.02	70.48	Entire AETD dataset: 10,065 manually labeled examples
Logistic Regression	65.62	63.51	58.42	69.57	
Data Augmentation	65.18	62.86	57.55	69.25	
Embedding Lookup	61.41	60.43	57.55	63.62	
BERT	78.49	78.86	78.34	79.38	
Stacking Ensemble	**80.26**	**80.77**	**80.94**	**80.59**	

The results obtained from both the AETD and ArPanEmo datasets clearly indicate that the stacking ensemble model effectively harnesses the strengths of all the base models when evaluated separately from self-training using the complete datasets of manually labeled data. When using transfer learning alone by fine-tuning the BERT-based model, a weighted average F1-score of 78.85% was achieved on the AETD dataset. On the other hand, training an emotion recognizer using the Naive Bayes algorithm and the AETD dataset resulted in a weighted average F1-score of 65.41%. However, the stacking ensemble of diverse models outperformed these individual models, achieving a weighted average F1-score of 80.77% on the AETD dataset.

Table 9. Performance of the stacking ensemble model and base classifiers on the ArPanEmo dataset.

Model	Accuracy	Weighted			Training Data (Size)
		F1	P	R	
Zero-Shot Classification	28.86	27.837773	27.27	28.43	No-training
Naive Bayes	59.85	61.58	63.94	59.39	Entire ArPanEmo dataset: 8901 manually labeled examples
Logistic Regression	58.20	59.90	62.06	57.89	
Data Augmentation	57.25	58.62	60.18	57.14	
Embedding Lookup	55.83	57.05	58.30	55.86	
BERT	66.95	67.64	68.64	66.67	
Stacking Ensemble	**69.08**	**69.93**	**71.46**	**68.47**	

The overall performance of the stacking ensemble model on ArPanEmo, as shown in Table 9, does not differ from that when using AETD dataset. The stacking ensemble model outperformed all six base classifiers, achieving a weighted average F1-score equal to 69.07%. This indicates that the meta-learner, which resulted from learning and combining the predictions of multiple base classifiers, yields superior results and improves predictive performance by leveraging the strengths of these base classifiers.

6.6.2. Analysis of Self-Training with Majority Voting

Regarding the second component of our proposed method, which is self-training with majority voting, as previously mentioned, we initiated the training process with only 10 labeled examples per emotion class from the benchmark datasets. In each iteration, additional data were automatically labeled by the base classifiers and incorporated into the training data through majority voting. Specifically, new data with its predictions were considered confident and included in the training dataset for subsequent iterations if at least 5 out of the 6 classifiers reached a consensus on its prediction. Tables 10 and 11 show the performance of self-training with majority voting on the AETD and ArPanEmo datasets.

Table 10. Performance of self-training with majority voting on the AETD dataset.

Model	Accuracy	Weighted			Training Data (Size)
		F1	P	R	
Zero-Shot Classification	34.03	30.93	42.67	34.03	No-training
Naive Bayes	69.67	68.94	69.8	69.67	AETD (80 manually labeled examples) + Automatically Labeled Data (22K)
Logistic Regression	66.1	63.63	63.65	66.09	
Data Augmentation	65.7	63.56	64.82	65.7	
Embedding Lookup	58.69	58.28	58.04	58.69	
BERT	**81.65**	**81.66**	**81.73**	**81.86**	
Self-training with Majority Voting	77.16	78.12	79.64	76.65	

Self-training with majority voting obviously demonstrated improved results compared to most individual base classifiers but fell short of the performance achieved by the fine-tuned BERT model. It yielded a weighted average F1-score of 78.12% on the AETD dataset, whereas the BERT model achieved a higher weighted average F1-score of 81.66%. The same results can be noticed when evaluating self-training on ArPanEmo, where the weighted average F1-score was 67.39%, while the fine-tuned BERT model outperformed it with a weighted average F1-score of 69.08%.

Table 11. Performance of self-training with majority voting on the ArPanEmo dataset.

Model	Accuracy	Weighted			Training Data (Size)
		F1	P	R	
Zero-Shot Classification	24.54	21.2	30.72	24.54	No-training
Naive Bayes	58.66	58.56	59.92	58.66	AETD (80 manually labeled examples) + Automatically Labeled Data (22K)
Logistic Regression	57.85	58.22	59.96	57.85	
Data Augmentation	56.91	56.75	56.12	56.91	
Embedding Lookup	49.79	49.65	50	49.79	
BERT	**69.02**	**69.08**	**71.02**	**69.02**	
Self-training with Majority Voting	66.41	67.39	68.97	65.87	

When comparing self-training with the stacking ensemble, it is evident that self-training with majority voting yielded a lower weighted average F1-score than the stacking ensemble model. This indicates the stacking ensemble model's ability to effectively leverage the strengths of the six base classifiers. However, it is worth noting that the F1-score of 78.12% achieved by self-training is still a good result, especially considering that self-training with majority voting initiates the training process with a limited number of manually labeled examples. On the other hand, the precision of self-training with majority voting is 79.64% on the AETD dataset, which is higher than the recall of 76.65%. This may be partially attributed to our strict condition for accepting a newly annotated label in majority voting, where it must receive the consensus of at least 5 out of the 6 votes from the base classifiers.

Self-training with majority voting obviously produces better results compared to five of the six base classifiers, falling about 1 to 3 points behind the fine-tuned BERT model. Nevertheless, self-training offers the advantage of gradually expanding the labeled data with minimal human intervention over iterations. On the other hand, the stacking ensemble of diverse models demonstrates more effective harnessing of the individual base models' strengths than self-training in terms of the final predictive performance. In our proposed method, we integrate the stacking ensemble model with self-training to exploit self-training's ability to automatically increase labeled data and the efficiency of the stacking ensemble model when combining predictions from a diverse set of base models. Our approach is specifically designed for low-resource settings, where labeled data are limited. Thus, the next section delves into the examination of our proposed approach in low-resource scenarios.

6.7. Low-Resource Scenarios (RQ3)

This section conducts a comparative analysis of our proposed method's performance in low-resource scenarios against the six base models. As previously explained, these base models include Naive Bayes, Logistic Regression, few-shot learning methods (i.e., nearest neighbor embedding lookup and data augmentation), a zero-shot classification model, and transfer learning (BERT-based models).

We initiated the training process of our proposed method with 80 labeled examples (10 examples per emotion class) from the AETD dataset. Table 12 shows the results of our proposed method on the AETD test set at the beginning stage using only the 10 examples per emotion class without going through iterations using self-training. The table also compares our proposed method with each base model in a low-resource setting.

The results show the impact of the first component of our proposed method, where a stacking ensemble model that combines different models is used on a limited amount of labeled data. We can see that our proposed method achieved the highest performance, with an accuracy of 39.96% and a weighted average F1-score of 39.47%. The second-best performing approach is the zero-shot model, which resulted in an accuracy of 37.5% and a weighted average F1-score of 33.27%. Next in line are the ML algorithms and few-shot learning methods. In the third, fourth, fifth, and sixth places are Logistic Regression, Naive Bayes, the embedding lookup method, and the data augmentation method, with weighted average F1-scores equal to 33.73%, 32.92%, 31.21%, and 21.88%, respectively. In terms of weighted average precision, the few-shot learning method, namely embedding lookup, achieved a higher result, equal to 35.58%, compared to the other few-shot learning method (i.e., data augmentation) and the other two ML algorithms (i.e., Naive Bayes and Logistic Regression).

Table 12. Performance results of models on the AETD dataset when trained using a small amount of labeled data (80 examples from AETD).

Method	Model	Accuracy	Weighted			Macro			Micro		
			F1	P	R	F1	P	R	F1	P	R
Zero-Shot Classification	XLM-RoBERTa-large-xnli model.	37.5	33.27	55.47	37.5	33.27	55.47	37.5	37.5	37.5	37.5
Machine Learning	Nive Bayes (NB)	34.18	32.92	33.39	34.18	32.74	33.31	33.94	34.18	34.18	34.18
	Logistic Regression (LR)	35.77	33.73	34.15	35.77	33.27	33.66	35.26	35.77	35.77	35.77
Few-Shot Learning	Data Augmentation	25	21.88	21.54	25	19.84	18.91	23.7	25	25	25
	Embedding Lookup	33.75	31.21	35.58	33.75	29.91	33.28	33.33	33.75	33.75	33.75
Transfer Learning	BERT	15.63	11.14	20.09	15.63	10.93	21.07	15.54	15.63	15.63	15.63
Our Proposed Approach		**39.96**	**39.47**	**40.12**	**39.96**	**39.45**	**40.45**	**39.58**	**39.96**	**39.96**	**39.96**

The lowest results were obtained when using transfer learning to fine-tune a model on a small amount of labeled data. The fine-tuned BERT model yielded a weighted average F1-score, precision, and recall of 11.14%, 20.09%, and 15.63%, respectively. The accuracy was also equal to 15.63%.

At the end of the training (i.e., after 20 iterations of self-training), the size of the training data had expanded from 80 to 22,544 examples. At this final stage, our proposed method still achieved the highest performance in all evaluation metrics compared to other approaches. Specifically, the accuracy, weighted average F1-score, precision, and recall were all equal to 83.15%, 83.19%, 83.31%, and 83.15%, respectively. Table 13 displays the results of our proposed method and other methods at the final stage after completing the 20 iterations on the test set.

The second-best performing model was not the zero-shot model, as it was in the first stage when starting with a low-resource setting (i.e., the start point with limited labeled data). Indeed, the second-best performing model in the final stage was the fine-tuned BERT-based model, which achieved an accuracy of 81.65%, along with a weighted average F1-score of 81.66%, precision of 81.73%, and recall of 81.86%. On the other hand, in the final stage, the zero-shot model ranked last, with an accuracy of 34.03% and a weighted average F1-score, precision, and recall of 30.93%, 42.67%, and 34.03%, respectively.

The performance of ML algorithms in the final stage was enhanced. Naive Bayes and Logistic Regression achieved weighted average F1-scores of 68.94% and 63.63%, respectively, showing an increase of 36.02% and 29.9% from the first stage. The few-shot models also demonstrated improvement after 20 iterations compared to the results they obtained in the first stage, achieving a weighted average F1-score of 58.28% and 63.56% for the embedding lookup and data augmentation methods, respectively.

Table 13. Performance results of models on the AETD dataset when trained using 22K labeled data (automatically labeled using self-training with majority voting).

Method	Model	Accuracy	Weighted			Macro			Micro		
			F1	P	R	F1	P	R	F1	P	R
Zero-Shot Classification	XLM-RoBERTa-large-xnli model.	34.03	30.93	42.67	34.03	32.44	42.6	37.06	34.03	34.03	34.03
Machine Learning	Nive Bayes (NB)	69.67	68.94	69.8	69.67	68.84	70.44	68.76	69.67	69.67	69.67
	Logistic Regression (LR)	66.1	63.63	63.65	66.09	63.63	68.76	64.9	66.1	66.1	66.1
Few-Shot Learning	Data Augmentation	65.7	63.56	64.82	65.7	63.46	65.26	64.87	65.7	65.7	65.7
	Embedding Lookup	58.69	58.28	58.04	58.69	58.18	58.04	58.49	58.69	58.69	58.69
Transfer Learning	BERT	81.65	81.66	81.73	81.86	81.86	82.01	81.78	81.65	81.65	81.65
Our Proposed Approach		**83.15**	**83.19**	**83.31**	**83.15**	**83.33**	**83.54**	**83.21**	**83.15**	**83.15**	**83.15**

Regarding the ArPanEmo dataset, all methods, including our proposed approach, were trained on a small number of labeled data (10 examples per class, resulting in 110 labeled examples from ArPanEmo). At this stage, no iterations were applied. Table 14 shows the results of our proposed method and the other methods when trained on a small number of labeled data.

Table 14. Performance results of models on ArPanEmo test set when trained using a small amount of labeled data (110 examples from ArPanEmo).

Method	Model	Accuracy	Weighted			Macro			Micro		
			F1	P	R	F1	P	R	F1	P	R
Zero-Shot Classification	XLM-RoBERTa-large-xnli model.	26.36	20.07	22.84	26.36	20.07	22.84	26.36	26.36	26.36	26.36
Machine Learning	Nive Bayes (NB)	24.79	24.88	26.6	24.79	22.45	23.84	22.79	24.79	24.79	24.79
	Logistic Regression (LR)	24.7	24.81	26.14	24.7	22.58	23.62	22.83	24.7	24.7	24.7
Few-Shot Learning	Data Augmentation	25.04	24.42	26.03	25.04	22.23	25.39	21.94	25.04	25.04	25.04
	Embedding Lookup	22.65	22.5	22.82	22.65	21.37	22.64	20.68	22.65	22.65	22.65
Transfer Learning	BERT	14.03	9.15	8.95	14.03	7.45	7.5	11.36	14.03	14.03	14.03
Our Proposed Approach		**31.53**	**31.24**	**31.47**	**31.53**	**29.41**	**30.16**	**29.33**	**31.53**	**31.53**	**31.53**

Our proposed method achieved the highest performance, with an accuracy of 31.53%. The weighted average F1-score, precision, and recall were 31.24%, 31.47%, and 31.53%, respectively. The second-best performing model was the zero-shot classification method, with an accuracy of 26.36%. The data augmentation method ranked third, with an accuracy of 25.04% and a weighted average F1-score of 24.42%. The model with the lowest performance when trained on a small number of labeled data was the fine-tuned BERT-based model, which achieved an accuracy of 14.03% and a weighted average F1-score of 9.15%.

Increasing the number of labeled training examples using self-training improved the overall performance of all models. At the final stage of training, when reaching the 20th iteration, the number of training examples increased to 21,135 labeled examples. The most significant positive impact was on the performance of our proposed approach, with accuracy increasing to 72.28%, while the weighted average F1-score, precision, and recall were 72.12%, 72.81%, and 72.28%, respectively. Table 15 displays the performance results of our proposed model on a large amount of automatically labeled data, as well as the results of the other models.

Table 15. Performance results of models on the ArPanEmo test set when trained using 21K labeled data (automatically labeled using self-training with majority voting).

Method	**Model**	**Accuracy**	**Weighted**			**Macro**			**Micro**		
			F1	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**
Zero-Shot Classification	XLM-RoBERTa-large-xnli model.	24.54	21.2	30.72	24.54	22.24	31.41	26.99	24.54	24.54	24.54
Machine Learning	Nive Bayes (NB)	58.66	58.56	59.92	58.66	57.01	58.66	56.72	58.66	58.66	58.66
	Logistic Regression (LR)	57.85	58.22	59.96	57.85	56.81	57.55	57.01	57.85	57.85	57.85
Few-Shot Learning	Data Augmentation	56.91	56.75	56.12	56.91	55.75	57.01	55.13	56.91	56.91	56.91
	Embedding Lookup	49.79	49.65	50	49.79	48.72	49.91	48.09	49.79	49.79	49.79
Transfer Learning	BERT	69.02	69.08	71.02	69.02	68.53	68.89	69.5	69.02	69.02	69.02
Our Proposed Approach		**72.28**	**72.12**	**72.81**	**72.28**	**70.63**	**72.47**	**69.83**	**72.28**	**72.28**	**72.28**

The second-best performing model was the fine-tuned BERT-based model on the 21K examples. The ML algorithms and few-shot methods ranked third, fourth, fifth, and sixth. The last model with the lowest values of evaluation metrics was the zero-shot model, which achieved an accuracy of 24.54%, while the weighted average F1-score, precision, and recall were 21.20%, 30.72%, and 24.54%, respectively.

7. Discussion

We propose a novel low-resource method for fine-grained Arabic emotion recognition consisting of two main components. The first component is related to training a stacking ensemble of various models using different learning paradigms. The second component involves self-training, which is responsible for the iterative scenario in which the number of labeled examples is gradually and carefully increased. Our overall approach comprising these two components outperformed various models that utilized different learning paradigms, achieving the highest weighted average F1-score of 83.19% and 72.12% on the AETD and ArPanEmo datasets, respectively. Additionally, each component has proved to be efficient in enhancing the performance of a low-resource, fine-grained model, as seen in Section 6.7.

When tested on a small number of labeled data and before applying self-training, the first component of our proposed approach, which entails creating a stacking ensemble of diverse base models, performed better compared to the base models individually. This was evident on two different datasets for Arabic emotion recognition: the AETD dataset and the ArPanEmo dataset, where only 10 examples per class were utilized to train and construct the models. Our proposed method achieved a weighted average F1-score of 39.47% and 31.24% when tested on the AETD and ArPanEmo datasets, respectively, while the best-performing base model obtained a weighted average F1-score of 33.27% and 24.88% on the AETD and ArPanEmo datasets, respectively.

In the second component of our proposed approach, which entails self-training, the number of labeled examples was carefully increased using majority voting of six models and a high threshold value equal to 5. That is, the final prediction was selected if at least five out of six base models voted for it. This automatic increase in labeled training examples through an iterative process positively affected our proposed method. Indeed, the performance of all base models improved, and our proposed method achieved the best results, with weighted average F1-scores of 83.19% and 72.12% on AETD and ArPanEmo, respectively.

To understand the behavior of each one of the six base models when trained on a small labeled dataset and when the labeled data gradually increased through self-training, we examined the relationships between the number of labeled training examples and the model's performance. We recorded the weighted average, macro average, and micro average F1-scores of our proposed approach and the other six base models during training with varying numbers of labeled examples. Figure 4 illustrates the relationships between the number of training samples and the models' performance when tested on the AETD dataset.

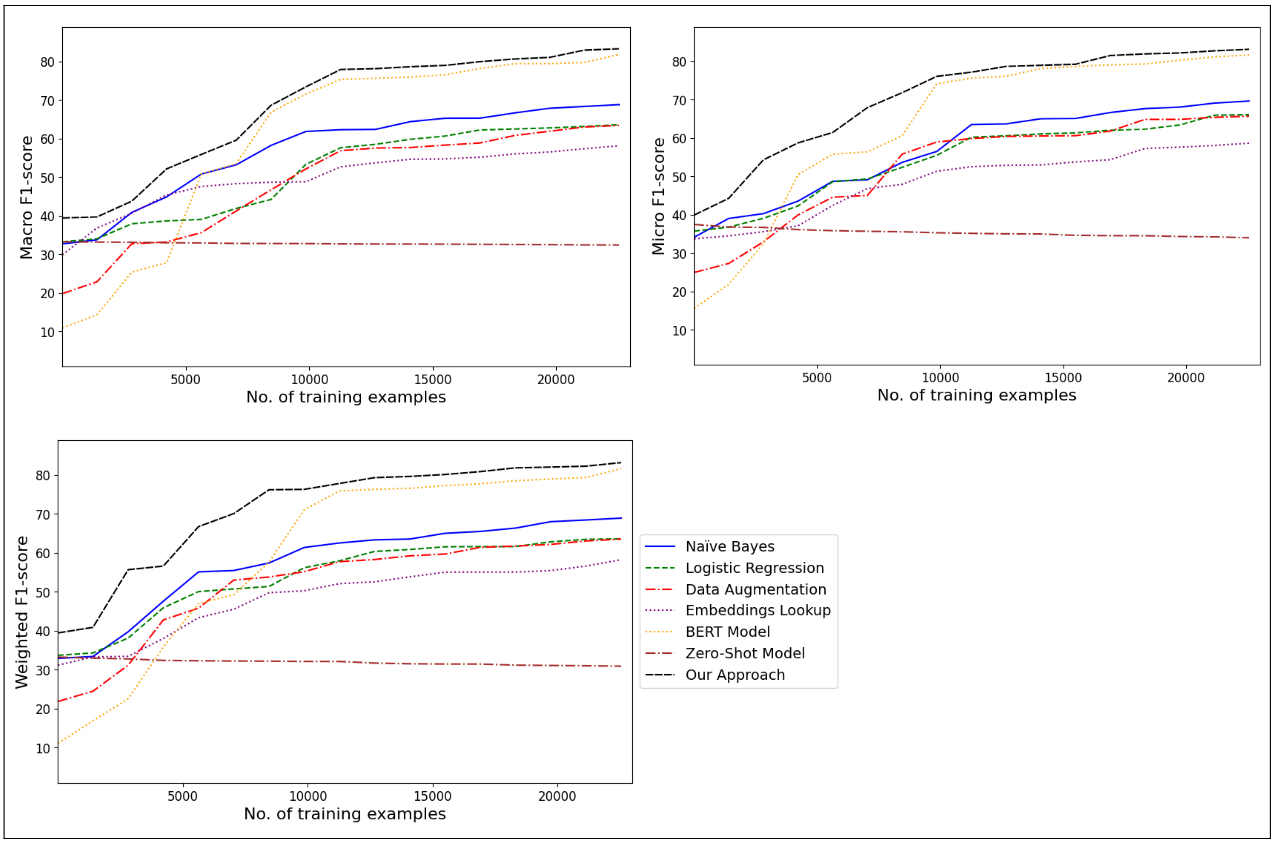

Figure 4. No. of training examples vs. F1-scores when tested on AETD.

From Figure 4, we can observe that the macro, micro, and weighted F1-scores all show improvements as the number of training examples increases. On the small labeled dataset, we noticed that the zero-shot model outperformed other methods, such as few-shot techniques and fine-tuned BERT-based model. However, as the number of training examples increased, the fine-tuned BERT model began to achieve better results than the zero-shot model and other models. ML algorithms and few-shot models performed similarly, and their performance improved with more training examples. However, this improvement was less than that seen in the BERT-based model, especially when the number of training examples reached thousands of labeled examples instead of just hundreds. In general, some models show strong performance with only a few labeled examples, such as the zero-shot model, while others show their strengths with larger number of labeled examples, such as the BERT-based model.

Our proposed method performs the best among all the other methods, leveraging the diverse strengths of its base models at both the initial stage of training when data are limited and at the advanced stage when a large number of labeled training data becomes available.

Figure 5 shows the relationships between the number of training examples and the performance of various models when tested on the ArPanEmo dataset.

Similar trends to those observed on the AETD dataset are also generally observed when using the ArPanEmo dataset, as depicted in Figure 5. We can observe that all models show improvement as the number of training examples increases. However, the extent of improvement varies depending on the model type. When starting with a small number of training examples (specifically, 10 examples per class from the ArPanEmo dataset), the zero-shot model emerges as the second-best performing model after our proposed model. As the number of training examples increases, the performance of the fine-tuned BERT-based model also increases accordingly. These findings align with the conclusions we drew earlier from our analysis of the AETD dataset in terms of the relationship between the number of training examples and the performance of various models.

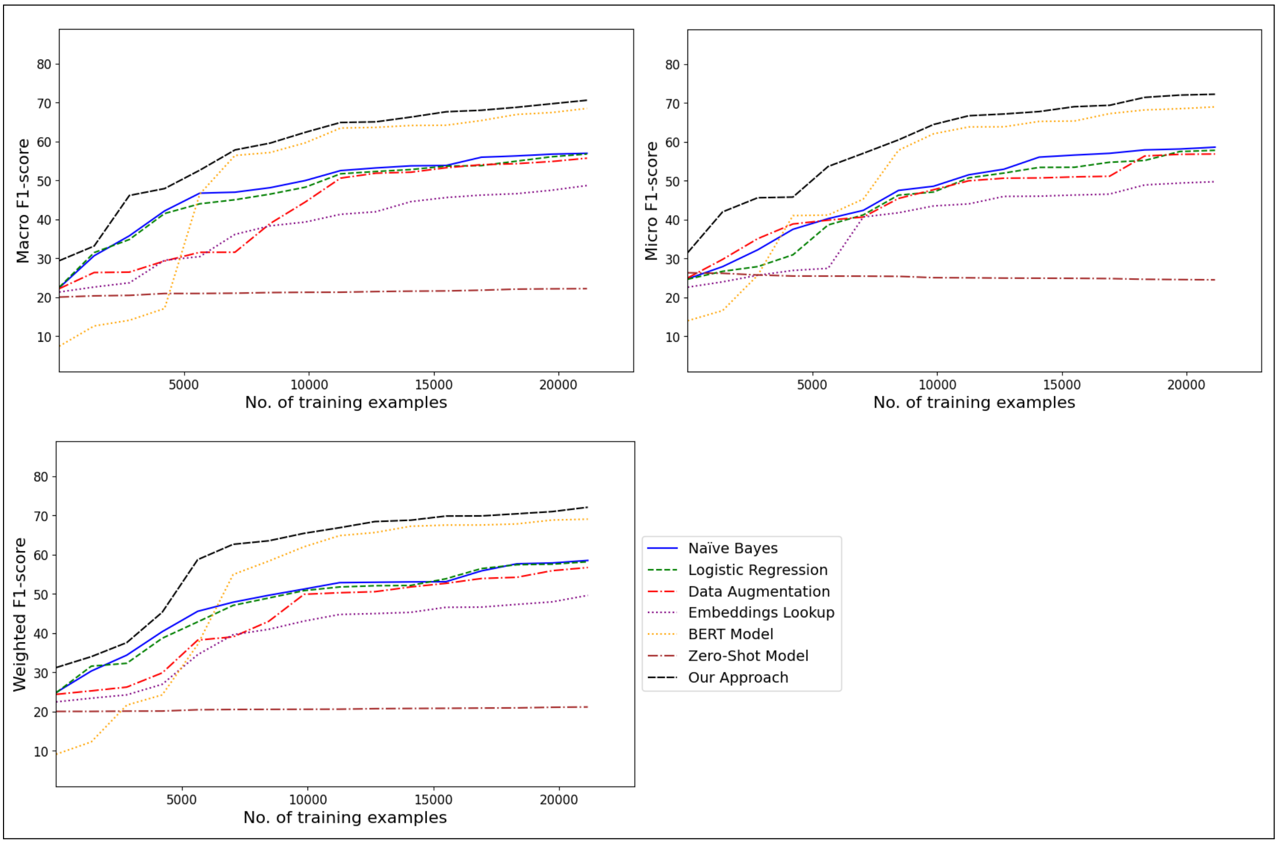

Figure 5. No. of training examples vs. F1-scores when tested on ArPanEmo.

As our proposed method depends on an iterative process to gradually generate labeled data, it mainly requires high computational resources. This computational cost is not only for generating new labeled data but also for training the models used in our proposed method. The diversity and the significant number of base models we employ are essential factors, but they come at the cost of time and complex calculations. Specifically, few-shot methods, such as data augmentation and embedding lookup, incur a computational cost that is directly proportional to the increasing data size in each iteration. This increase in data size and the computational cost for the few-shot methods does not yield a proportionate increase in the performance of their trained models. This is evidently illustrated in Figures 4 and 5, where it can be seen that other methods, such as the fine-tuned BERT model and conventional machine learning methods, consistently outperform the few-shot methods as the data size increases. Furthermore, it is crucial to carefully select data based on the application's specific needs. The base models are trained on labeled data generated from these data and may inherit any biases present in the data, such as skewed representations of certain classes.

8. Conclusions and Future Works

The need for specialized and nuanced classification has drawn attention to the challenge of creating fine-grained text emotion recognition models with high performance, particularly when confronted with limited labeled data. In response to this challenge, our study proposes an innovative approach that addresses low-resource, fine-grained Arabic emotion recognition. Our method entails an iterative framework comprising a stacking ensemble model and a self-training process. It eliminates the need for an extensive initial labeled dataset, progressively generating labeled data through a careful process that uses the majority voting of a set of diverse models. The stacking ensemble model learns to combine the prediction of various base models employing different learning paradigms, which helps to gain an ensemble model with better performance, exploiting the strengths of each base model. The utilized base models include few-shot models, a zero-shot classification model, machine learning models, and a BERT-based model.

The diverse models were chosen carefully as some of them were specifically designed for low-resource data, such as the few-shot methods and zero-shot classification, while others performed well on thousands of labeled data, such as the BERT-based model and ML algorithms. The low-resource models helped our approach in the first stage of training, while the ML and BERT-based algorithms were helpful in the late stage of the iterative framework after the number of labeled training examples increased to the thousands.

Through extensive experiments, we thoroughly assessed the individual performance of each base model for fine-grained, low-resource Arabic emotion recognition. The experimental results showed that our approach surpassed the individual performance of each base model when using a small number of labeled data. Although our approach to Arabic emotion recognition started with a small number of labeled data, it outperformed existing state-of-the-art Arabic emotion recognition models in the literature. This achievement is highlighted by our approach achieving the highest weighted average F1-scores of 83.19% and 72.12% on the AETD and ArPanEmo benchmark datasets, respectively.

For future research, exploring various semi-supervised learning approaches, such as Expectation Maximization, Mean Teacher, and graph-based methods, could be promising alternatives to the self-training process utilized in our proposed method. In addition, examining various learning paradigms for the base models could be beneficial, including, for example, exploring alternative few-shot methods, such as prototypical networks instead of the data augmentation and embedding lookup utilized in our proposed method.

Future research directions could also focus on proposing methods that address low-resource fine-grained text classification while reducing computational costs. Several few-shot learning methods, including data augmentation and embedding lookup, involve high computational costs, prompting the need for further investigation into cost-effective alternatives and comparisons between different methods.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets utilized in the experiments of this study can be found here: https://data.mendeley.com/datasets/d9yy8w52ns/2 (accessed at 30 July 2023) and https://github.com/amrmalkhatib/Emotional-Tone (accessed at 5 May 2022).

Conflicts of Interest: The author declares no conflict of interest.

References

1. Cortal, G.; Finkel, A.; Paroubek, P.; Ye, L. Emotion Recognition based on Psychological Components in Guided Narratives for Emotion Regulation. In Proceedings of the 7th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature, Dubrovnik, Croatia, 5 May 2023; pp. 72–81.
2. Sintsova, V.; Musat, C.; Pu, P. Fine-Grained Emotion Recognition in Olympic Tweets Based on Human Computation. In Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, Atlanta, GA, USA, 14 June 2013; pp. 12–20.
3. Sebe, N.; Cohen, I.; Huang, T.S. Multimodal emotion recognition. In *Handbook of Pattern Recognition and Computer Vision*; World Scientific: Singapore, 2005; pp. 387–409.
4. Chuang, Z.J.; Wu, C.H. Multi-Modal Emotion Recognition from Speech and Text. *Int. J. Comput. Linguist. Chin. Lang. Process.* **2004** , *9*, 45–62.
5. Aman, S.; Szpakowicz, S. Using Roget's Thesaurus for Fine-grained Emotion Recognition. In Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-I, Hyderabad, India, 7–12 January 2008.

6. Jang, E.H.; Park, B.J.; Kim, S.H.; Chung, M.A.; Park, M.S.; Sohn, J.H. Emotion classification based on bio-signals emotion recognition using machine learning algorithms. In Proceedings of the 2014 International Conference on Information Science, Electronics and Electrical Engineering, Sapporo, Japan, 26–28 April 2014; Volume 3, pp. 1373–1376.
7. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
8. Kane, A.; Patankar, S.; Khose, S.; Kirtane, N. Transformer based ensemble for emotion detection. In Proceedings of the 12th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis, Dublin, Ireland, 26 May 2022; pp. 250–254. [CrossRef]
9. Nedilko, A. Generative Pretrained Transformers for Emotion Detection in a Code-Switching Setting. In Proceedings of the 13th Workshop on Computational Approaches to Subjectivity, Sentiment, & Social Media Analysis, Toronto, ON, Canada, 14 July 2023; pp. 616–620.
10. Zou, X.; Yan, Y.; Xue, J.H.; Chen, S.; Wang, H. When Facial Expression Recognition Meets Few-Shot Learning: A Joint and Alternate Learning Framework. *Proc. AAAI Conf. Artif. Intell.* **2022**, *36*, 5367–5375. [CrossRef]
11. Chen, X.; Zheng, X.; Sun, K.; Liu, W.; Zhang, Y. Self-supervised vision transformer-based few-shot learning for facial expression recognition. *Inf. Sci.* **2023**, *634*, 206–226. [CrossRef]
12. Jacovi, A.; Shalom, O.S.; Goldberg, Y. Understanding Convolutional Neural Networks for Text Classification. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018.
13. Yu, Y.; Si, X.; Hu, C.; Zhang, J. A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. *Neural Comput.* **2019**, *31*, 1235–1270. [CrossRef] [PubMed]
14. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the NAACL-HLT, Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186.
15. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving Language Understanding by Generative Pre-Training. 2018. Available online: https://www.mikecaptain.com/resources/pdf/GPT-1.pdf (accessed on 20 June 2023).
16. Pitaloka, D.A.; Wulandari, A.; Basaruddin, T.; Liliana, D.Y. Enhancing CNN with preprocessing stage in automatic emotion recognition. *Procedia Comput. Sci.* **2017**, *116*, 523–529. [CrossRef]
17. Aslan, M. CNN based efficient approach for emotion recognition. *J. King Saud Univ.-Comput. Inf. Sci.* **2022**, *34*, 7335–7346. [CrossRef]
18. Pereira, P.; Moniz, H.; Dias, I.; Carvalho, J.P. Context-Dependent Embedding Utterance Representations for Emotion Recognition in Conversations. In Proceedings of the 13th Workshop on Computational Approaches to Subjectivity, Sentiment, & Social Media Analysis, Toronto, ON, Canada, 13 January 2023; pp. 228–236.
19. Chu, I.H.; Chen, Z.; Yu, X.; Han, M.; Xiao, J.; Chang, P. Self-supervised Cross-modal Pretraining for Speech Emotion Recognition and Sentiment Analysis. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, 7–11 December 2022; pp. 5105–5114. [CrossRef]
20. Karna, M.; Juliet, D.S.; Joy, R.C. Deep learning based text emotion recognition for chatbot applications. In Proceedings of the 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI) (48184), Tirunelveli, India, 16–18 April 2020; pp. 988–993.
21. Khanpour, H.; Caragea, C. Fine-Grained Emotion Detection in Health-Related Online Posts. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 7 June 2018; pp. 1160–1166. [CrossRef]
22. Alqahtani, G.; Alothaim, A. Emotion Analysis of Arabic Tweets: Language Models and Available Resources. *Front. Artif. Intell.* **2022**, *5*, 843038. [CrossRef]
23. Demszky, D.; Movshovitz-Attias, D.; Ko, J.; Cowen, A.; Nemade, G.; Ravi, S. GoEmotions: A Dataset of Fine-Grained Emotions. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 10 July 2020; pp. 4040–4054.
24. Sosea, T.; Caragea, C. Canceremo: A dataset for fine-grained emotion detection. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16–20 November 2020; pp. 8892–8904.
25. Qin, X.; Wu, Z.; Zhang, T.; Li, Y.; Luan, J.; Wang, B.; Wang, L.; Cui, J. BERT-ERC: Fine-tuning BERT is enough for emotion recognition in conversation. In Proceedings of the AAAI Conference on Artificial Intelligence, Arlington, VA, USA, 7–14 February 2023; Volume 37, pp. 13492–13500.
26. Zygadło, A.; Kozłowski, M.; Janicki, A. Text-Based emotion recognition in English and Polish for therapeutic chatbot. *Appl. Sci.* **2021**, *11*, 10146. [CrossRef]
27. Chowdhury, M.S.M.; Pal, B. BERT-Based Emotion Classification Approach with Analysis of COVID-19 Pandemic Tweets. In *Applied Informatics for Industry 4.0*; Chapman and Hall/CRC: Boca Raton, FL, USA, 2023; pp. 109–121.
28. Adoma, A.F.; Henry, N.M.; Chen, W. Comparative analyses of bert, roberta, distilbert, and xlnet for text-based emotion recognition. In Proceedings of the 2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), Chengdu, China, 17–19 December 2021; pp. 117–121.
29. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.R.; Le, Q.V. Xlnet: Generalized autoregressive pretraining for language understanding. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 5753–5763.
30. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
31. Kumar, P.; Raman, B. A BERT based dual-channel explainable text emotion recognition system. *Neural Netw.* **2022**, *150*, 392–407. [CrossRef]

32. Alm, C.O.; Roth, D.; Sproat, R. Emotions from text: Machine learning for text-based emotion prediction. In Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, Vancouver, BC, Canada, 6–8 October 2005; pp. 579–586.
33. Li, Y.; Su, H.; Shen, X.; Li, W.; Cao, Z.; Niu, S. DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset. In Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Taipei, Taiwan, 27 November–1 December 2017; pp. 986–995.
34. Oberländer, L.A.M.; Klinger, R. An analysis of annotated corpora for emotion classification in text. In Proceedings of the 27th International Conference on Computational Linguistics, Santa Fe, NM, USA, 20–26 August 2018; pp. 2104–2119.
35. Abd Al-Aziz, A.M.; Gheith, M.; Eldin, A.S. Lexicon based and multi-criteria decision making (MCDM) approach for detecting emotions from Arabic microblog text. In Proceedings of the 2015 First International Conference on Arabic Computational Linguistics (ACLing), Cairo, Egypt, 17–20 April 2015; pp. 100–105.
36. Al-A'abed, M.; Al-Ayyoub, M. A lexicon-based approach for emotion analysis of arabic social media content. In Proceedings of the The International Computer Sciences and Informatics Conference (ICSIC), Amman, Jordan, 12–13 January 2016; pp. 343–351.
37. El Gohary, A.F.; Sultan, T.I.; Hana, M.A.; El Dosoky, M. A computational approach for analyzing and detecting emotions in Arabic text. *Int. J. Eng. Res. Appl.* **2013**, *3*, 100–107.
38. Hussien, W.A.; Tashtoush, Y.M.; Al-Ayyoub, M.; Al-Kabi, M.N. Are emoticons good enough to train emotion classifiers of arabic tweets? In Proceedings of the 2016 7th International Conference on Computer Science and Information Technology (CSIT), Amman, Jordan, 13–14 July 2016; pp. 1–6.
39. Al-Khatib, A.; El-Beltagy, S.R. Emotional tone detection in arabic tweets. In Proceedings of the International Conference on Computational Linguistics and Intelligent Text Processing, Budapest, Hungary, 17–23 April 2017; pp. 105–114.
40. Mulki, H.; Ali, C.B.; Haddad, H.; Babaoğlu, I. Tw-star at semeval-2018 task 1: Preprocessing impact on multi-label emotion classification. In Proceedings of the 12th International Workshop on Semantic Evaluation, New Orleans, LA, USA, 8 January 2018; pp. 167–171.
41. Boser, B.E.; Guyon, I.M.; Vapnik, V.N. A training algorithm for optimal margin classifiers. In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, Pittsburgh, PA, USA, 27–29 July 1992; pp. 144–152.
42. Mohammad, S.; Bravo-Marquez, F.; Salameh, M.; Kiritchenko, S. Semeval-2018 task 1: Affect in tweets. In Proceedings of the 12th International Workshop on Semantic Evaluation, New Orleans, LA, USA, 5–6 June 2018; pp. 1–17.
43. Badaro, G.; El Jundi, O.; Khaddaj, A.; Maarouf, A.; Kain, R.; Hajj, H.; El-Hajj, W. EMA at SemEval-2018 task 1: Emotion mining for Arabic. In Proceedings of the 12th International Workshop on Semantic Evaluation, New Orleans, LA, USA, 5–6 June 2018; pp. 236–244.
44. Soliman, A.B.; Eissa, K.; El-Beltagy, S.R. Aravec: A set of arabic word embedding models for use in arabic nlp. *Procedia Comput. Sci.* **2017**, *117*, 256–265. [CrossRef]
45. Bojanowski, P.; Grave, E.; Joulin, A.; Mikolov, T. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguist.* **2017**, *5*, 135–146. [CrossRef]
46. Abdullah, M.; Shaikh, S. Teamuncc at semeval-2018 task 1: Emotion detection in english and arabic tweets using deep learning. In Proceedings of the 12th International Workshop on Semantic Evaluation, New Orleans, LA, USA, 5–6 June 2018; pp. 350–357.
47. Alswaidan, N.; Menai, M.E.B. Hybrid feature model for emotion recognition in Arabic text. *IEEE Access* **2020**, *8*, 37843–37854. [CrossRef]
48. Eisner, B.; Rocktäschel, T.; Augenstein, I.; Bošnjak, M.; Riedel, S. emoji2vec: Learning emoji representations from their description. *arXiv* **2016**, arXiv:1609.08359.
49. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
50. Grave, É.; Bojanowski, P.; Gupta, P.; Joulin, A.; Mikolov, T. Learning Word Vectors for 157 Languages. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan, 7–12 May 2018.
51. Antoun, W.; Baly, F.; Hajj, H. AraBERT: Transformer-based Model for Arabic Language Understanding. In Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection, Marseille, France, 12 May 2020; pp. 9–15.
52. Safaya, A.; Abdullatif, M.; Yuret, D. Kuisail at semeval-2020 task 12: Bert-cnn for offensive speech identification in social media. In Proceedings of the Fourteenth Workshop on Semantic Evaluation, Barcelona, Spain, 12–13 December 2020; pp. 2054–2059.
53. Antoun, W.; Baly, F.; Hajj, H. AraELECTRA: Pre-Training Text Discriminators for Arabic Language Understanding. In Proceedings of the Sixth Arabic Natural Language Processing Workshop, Kiev, Ukraine, 19 April 2021; pp. 191–195.
54. Antoun, W.; Baly, F.; Hajj, H. AraGPT2: Pre-Trained Transformer for Arabic Language Generation. In Proceedings of the Sixth Arabic Natural Language Processing Workshop, Kiev, Ukraine, 19 April 2021; pp. 196–207.
55. Abdelali, A.; Hassan, S.; Mubarak, H.; Darwish, K.; Samih, Y. Pre-training bert on arabic tweets: Practical considerations. *arXiv* **2021**, arXiv:2102.10684.
56. Abdul-Mageed, M.; Elmadany, A.; Nagoudi, E.M.B. ARBERT & MARBERT: Deep Bidirectional Transformers for Arabic. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Online, 1–6 August 2021; pp. 7088–7105. [CrossRef]

57. ALTHOBAITI, M.J. Emotion Recognition in Arabic: A Bert-Based Transfer Learning Approach Leveraging Semantic Information of Online Comments. *J. Theor. Appl. Inf. Technol.* **2023**, *101*, 3270–3282.
58. Abdul-Mageed, M.; AlHuzli, H.; DuaaAbu Elhija, M.D. Dina: A multi-dialect dataset for arabic emotion analysis. In Proceedings of the The 2nd Workshop on Arabic Corpora and Processing Tools, Portorož, Slovenia, 24 May 2016; p. 29.
59. Alhuzali, H.; Abdul-Mageed, M.; Ungar, L. Enabling deep learning of emotion with first-person seed expressions. In Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media, New Orleans, LA, USA, 6 June 2018; pp. 25–35.
60. Almahdawi, A.J.; Teahan, W.J. A new arabic dataset for emotion recognition. In *Proceedings of the Intelligent Computing-Proceedings of the Computing Conference*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 200–216.
61. Yang, Q.; Alamro, H.; Albaradei, S.; Salhi, A.; Lv, X.; Ma, C.; Alshehri, M.; Jaber, I.; Tifratene, F.; Wang, W.; et al. Senwave: Monitoring the global sentiments under the COVID-19 pandemic. *arXiv* **2020**, arXiv:2006.10842.
62. Althobaiti, M.J. An open-source dataset for arabic fine-grained emotion recognition of online content amid COVID-19 pandemic. *Data Brief* **2023**, *51*, 109745. [CrossRef]
63. Abdul-Mageed, M.; Zhang, C.; Hashemi, A.; Nagoudi, E.M.B. AraNet: A Deep Learning Toolkit for Arabic Social Media. In Proceedings of the LREC 2020 Workshop Language Resources and Evaluation Conference, Marseille, France, 11–16 May 2020; p. 16.
64. Luo, Q.; Liu, L.; Lin, Y.; Zhang, W. Don't miss the labels: Label-semantic augmented meta-learner for few-shot text classification. In Proceedings of the Findings of the Association for Computational Linguistics: ACL-IJCNLP, Online, 1–6 August 2021; pp. 2773–2782.
65. Zhou, Z.H. *Ensemble Methods: Foundations and Algorithms*; CRC Press: Boca Raton, FL, USA, 2012.
66. Tian, P.; Yu, H. Can we improve meta-learning model in few-shot learning by aligning data distributions? *Knowl.-Based Syst.* **2023**, *277*, 110800. [CrossRef]
67. Sahu, B.; Agrawal, S.; Dey, H.; Raj, C. Performance Analysis of State-of-the-Art Classifiers and Stack Ensemble Model for Liver Disease Diagnosis. In *Biologically Inspired Techniques in Many Criteria Decision Making: Proceedings of BITMDM 2021*; Springer: Singapore, 2022; pp. 95–105.
68. Sagi, O.; Rokach, L. Ensemble learning: A survey. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2018**, *8*, e1249. [CrossRef]
69. Althobaiti, M.J. Country-level Arabic Dialect Identification Using Small Datasets with Integrated Machine Learning Techniques and Deep Learning Models. In Proceedings of the Sixth Arabic Natural Language Processing Workshop, Kyiv, Ukraine (Virtual), 19 April 2021; pp. 265–270.
70. Geetha, M.; Renuka, D.K. Improving the performance of aspect based sentiment analysis using fine-tuned Bert Base Uncased model. *Int. J. Intell. Netw.* **2021**, *2*, 64–69. [CrossRef]
71. Xu, X.; Wang, G.; Kim, Y.B.; Lee, S. AugNLG: Few-shot Natural Language Generation using Self-trained Data Augmentation. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Online, 1–6 August 2021; pp. 1183–1195. [CrossRef]
72. Wei, J.; Zou, K. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 6382–6388.
73. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1877–1901.
74. Zhang, Y.; Yuan, C.; Wang, X.; Bai, Z.; Liu, Y. Learn to adapt for generalized zero-shot text classification. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Dublin, Ireland, 22–27 May 2022; pp. 517–527.
75. Feng, S.Y.; Gangal, V.; Wei, J.; Chandar, S.; Vosoughi, S.; Mitamura, T.; Hovy, E. A Survey of Data Augmentation Approaches for NLP. In Proceedings of the Findings of the Association for Computational Linguistics: ACL-IJCNLP, Online, 1–6 August 2021; pp. 968–988.
76. Gao, F.; Zhu, J.; Wu, L.; Xia, Y.; Qin, T.; Cheng, X.; Zhou, W.; Liu, T.Y. Soft Contextual Data Augmentation for Neural Machine Translation. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 5539–5544. [CrossRef]
77. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30* , 5998–6008.
78. Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv* **2019**, arXiv:1910.03771.
79. Tunstall, L.; Von Werra, L.; Wolf, T. *Natural Language Processing with Transformers*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2022.
80. Conneau, A.; Rinott, R.; Lample, G.; Williams, A.; Bowman, S.R.; Schwenk, H.; Stoyanov, V. XNLI: Evaluating Cross-lingual Sentence Representations. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Brussels, Belgium, 31 October–4 November 2018.
81. Williams, A.; Nangia, N.; Bowman, S. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), New Orleans, LA, USA, 1–6 June 2018; pp. 1112–1122.

82. Howard, J.; Ruder, S. Universal Language Model Fine-tuning for Text Classification. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, Australia, 15–20 July 2018; pp. 328–339.
83. Herzig, J.; Nowak, P.K.; Müller, T.; Piccinno, F.; Eisenschlos, J.M. TaPas: Weakly supervised table parsing via pre-training. *arXiv* **2020**, arXiv:2004.02349.
84. Althobaiti, M.J. BERT-based Approach to Arabic Hate Speech and Offensive Language Detection in Twitter: Exploiting Emojis and Sentiment Analysis. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*, 972–980. [CrossRef]
85. Keya, A.J.; Wadud, M.A.H.; Mridha, M.; Alatiyyah, M.; Hamid, M.A. AugFake-BERT: Handling imbalance through augmentation of fake news using BERT to enhance the performance of fake news classification. *Appl. Sci.* **2022**, *12*, 8398. [CrossRef]
86. Manning, C.; Schutze, H. *Foundations of Statistical Natural Language Processing*; MIT Press: Cambridge, MA, USA, 1999.
87. Havrlant, L.; Kreinovich, V. A simple probabilistic explanation of term frequency-inverse document frequency (tf-idf) heuristic (and variations motivated by this explanation). *Int. J. Gen. Syst.* **2017**, *46*, 27–36. [CrossRef]
88. Murphy, K.P. Naive bayes classifiers. *Univ. Br. Columbia* **2006**, *18*, 1–8.
89. Webb, G.I.; Keogh, E.; Miikkulainen, R. Naïve Bayes. *Encycl. Mach. Learn.* **2010**, *15*, 713–714.
90. Kim, S.B.; Han, K.S.; Rim, H.C.; Myaeng, S.H. Some effective techniques for naive bayes text classification. *IEEE Trans. Knowl. Data Eng.* **2006**, *18*, 1457–1466.
91. Chen, J.; Huang, H.; Tian, S.; Qu, Y. Feature selection for text classification with Naïve Bayes. *Expert Syst. Appl.* **2009**, *36*, 5432–5435. [CrossRef]
92. Burges, C.J. A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov.* **1998**, *2*, 121–167. [CrossRef]
93. Nigam, K.; McCallum, A.K.; Thrun, S.; Mitchell, T. Text classification from labeled and unlabeled documents using EM. *Mach. Learn.* **2000**, *39*, 103–134. [CrossRef]
94. Kleinbaum, D.G.; Klein, M.; Pryor, E.R. *Logistic Regression: A Self-Learning Text*; Springer: Berlin/Heidelberg, Germany, 2002; Volume 94.
95. Hastie, T.; Tibshirani, R.; Friedman, J.H.; Friedman, J.H. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*; Springer: Berlin/Heidelberg, Germany, 2009; Volume 2.
96. Zhu, X.J. *Semi-Supervised Learning Literature Survey*; University of Wisconsin-Madisoa: Madison, WI, USA, 2005.
97. Li, Z.; Guo, Q.; Pan, Y.; Ding, W.; Yu, J.; Zhang, Y.; Liu, W.; Chen, H.; Wang, H.; Xie, Y. Multi-level correlation mining framework with self-supervised label generation for multimodal sentiment analysis. *Inf. Fusion* **2023**, *99*, 101891. [CrossRef]
98. Nabil, M.; Aly, M.; Atiya, A. Astd: Arabic sentiment tweets dataset. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 2515–2519.
99. Althobaiti, M.; Kruschwitz, U.; Poesio, M. AraNLP: A Java-based Library for the Processing of Arabic Text. In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), Association for Computational Linguistics, Reykjavik, Iceland, 26–31 May 2014; pp. 4134–4138.
100. Johnson, J.; Douze, M.; Jégou, H. Billion-scale similarity search with gpus. *IEEE Trans. Big Data* **2019**, *7*, 535–547. [CrossRef]
101. Tiedemann, J.; Thottingal, S. OPUS-MT – Building open translation services for the World. In Proceedings of the 22nd Annual Conference of the European Association for Machine Translation, Lisbon, Portugal, 3–5 November 2020; pp. 479–480.
102. Tiedemann, J. The Tatoeba Translation Challenge – Realistic Data Sets for Low Resource and Multilingual MT. In Proceedings of the Fifth Conference on Machine Translation, Online, 19–20 November 2020; pp. 1174–1182.
103. Conneau, A.; Khandelwal, K.; Goyal, N.; Chaudhary, V.; Wenzek, G.; Guzmán, F.; Grave, É.; Ott, M.; Zettlemoyer, L.; Stoyanov, V. Unsupervised Cross-lingual Representation Learning at Scale. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 8440–8451.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.