DeepEmotex: Classifying Emotion in Text Messages using Deep Transfer Learning

Maryam Hasan
Department of Computer Science
Worcester Polytechnic Institute
and San Francisco State University
Email: mhasan@wpi.edu

Elke Rundensteiner
Department of Computer Science
and Data Science
Worcester Polytechnic Institute
Email: rundenst@wpi.edu

Emmanuel Agu
Department of Computer Science
Worcester Polytechnic Institute
Email: agu@wpi.edu

***Abstract*—Transfer learning has been widely used in natural language processing through deep pretrained language models, such as Bidirectional Encoder Representations from Transformers and Universal Sentence Encoder. Despite the great success, language models get overfitted when applied to small datasets and are prone to forgetting when fine-tuned with a classifier. To remedy this problem of forgetting in transferring deep pretrained language models from one domain to another domain, existing efforts explore fine-tuning methods to forget less. We propose DeepEmotex an effective sequential transfer learning method to detect emotion in text. To avoid forgetting problem, the fine-tuning step is instrumented by a large amount of emotion-labeled data collected from Twitter.**

We conduct an experimental study using both curated Twitter data sets and benchmark data sets. DeepEmotex models achieve over 91% accuracy for multi-class emotion classification on test dataset. We evaluate the performance of the fine-tuned DeepEmotex models in classifying emotion in EmoInt and Stimulus benchmark datasets. The models correctly classify emotion in 73% of the instances in the benchmark datasets. The proposed DeepEmotex-BERT model outperforms Bi-LSTM result on the benchmark datasets by 23%. We also study the effect of the size of the fine-tuning dataset on the accuracy of our models. Our evaluation results show that fine-tuning with a large set of emotion-labeled data improves both the robustness and effectiveness of the resulting target task model.

***Index Terms*—emotion detection, deep learning, transfer learning, neural networks.**

I. Introduction

Detecting emotions in text is a challenging problem because of the semantic ambiguity of the emotion expression in text and fuzzy boundaries of emotion classes [1]. Moreover, the context can completely change the emotion for a sentence as compared to perceived emotion when the sentence is evaluated standalone. For example, the sentence "I started crying when I realized!" will be perceived as a sad feeling, however considering it in the context "I just qualified for the scholarship. I started crying when I realized!", it turns out to be a happy emotion. Similarly, the sentence "My cat gets angry when I work from home!" within the context "My cat gets angry when I work from home. It is funny!". Another example is the sentence "Try to do that again!" is very likely to be perceived as no-emotion, however a majority will judge it as an Angry feeling with the context "How dare you make fun of me like that! Try to do that again!". Therefore, understanding the context is important to detect emotion in text.

Feature selection and representation are important tasks for emotion classification in texts due to the high dimensionality of text features and the existence of irrelevant features. The performance of classification algorithms strongly depends on feature selection and representation [2].

Supervised learning methods have proven to be promising in emotion classification. However, these methods are domain dependent, which means that a model built on one domain (e.g., messages on a specific topic or event) may perform poorly on another domain. The reason is that rather distinct words and phrases domain-specific words may be used to express emotion in different domains. Table I shows common keywords for two different domains, death of George Floyd and Covid-19 pandemic. For example, keywords *justice, protest, violent, murder, and racism* are domain-specific words characteristic of the first domain whereas keywords such as *death, disease, caugh, and fever* are Covid19-specific keywords. Due to the mismatch of common keywords between these different domains, an emotion classifier trained on one domain may not work well when directly applied to another domain. Therefore, cross-domain emotion classification methods are desirable. Such methods reduce domain dependency and the costs and human labor required for manually labeling a sufficient number of example texts required for training supervised learning models by transferring knowledge from related domains to the domain of interest [3].

For decades, supervised learning methods solving NLP problems have been based on shallow models (e.g., SVM and logistic regression) trained on very high dimensional, sparse and hand-crafted features [4]. In the last few years, neural networks based on a rich variety of representations have been producing much superior to prior winning results on various NLP tasks [4]. This trend is sparked by the success of distributed word embeddings including Word2Vec and GloVe [5], [6].

Deep neural network-based approaches have achieved remarkable successes on text classification tasks by being pretrained on huge volumes of text archives such as Wikipedia. Deep learning models automatically learn multiple layers of low dimensional feature representations and thus reduce the

TABLE I
EXAMPLE KEYWORDS OF DIFFERENT DOMAINS

Death of George Floyd	attack, violence, protest, curfew, death, murder, racism, black lives
Covid-19 pandemic	fever, death, patient, disease, vaccine, mask, immunity, pandemic

need for hand-crafted features [4]. Deep learning methods have been utilized in learning word representations through neural language models [5]. Along with the success of deep learning in many other application domains, deep learning has also become popular for solving sentiment and emotion classification in recent years [4].

Different neural network architectures have emerged to deal with challenges in NLP tasks, such as RNNs, LSTM, BiLSTM. More recently, the Transformers architecture [7] has been shown to have extremely promising results for NLP tasks. Transformers language models have shown better language understanding abilities that allow them to achieve state-of-the-art results for many different NLP tasks.

While deep learning models have achieved state-of-the-art results on text classification tasks, these models are trained from scratch, requiring huge input datasets, and days to converge [8]. Instead of learning from scratch, transfer learning can be used to transfer knowledge from a general-purpose domain and task into a more specialized target domain and task [9], [10], [11]. In fact, transfer learning in many cases can achieve or even exceed the performance of traditional deep learning models trained for the particular task from scratch, yet while only requiring a much smaller set of labeled examples for the fine-tuning to the target task [8]. Transfer learning can also be beneficial for sentiment and emotion classification in text.

Transfer learning has been widely used in natural language processing (NLP) through pretrained Language Models (LMs) [12]. Deep pretrained Language Models, such as Universal Sentence Encoder (USE) [13] and Bidirectional Encoder Representations from Transformers (BERT) [14], have been widely used in Natural Language Processing (NLP). Despite the great success, LMs get overfitted to small datasets and are prone to catastrophic forgetting when fine-tuned with a classifier [8]. To remedy the catastrophic forgetting in transferring deep pretrained LMs, existing efforts mainly explore fine-tuning tricks to forget less. Fine-tuning experiments rely on the pretrained language model parameters. During target task training, the language model must adapt enough to be able to solve the target task involving a different input distribution and output label space than the pretrained model. The adaptation step must also avoid overfitting or forgetting of what was learned during pretraining [15].

We propose DeepEmotex, an effective sequential transfer learning method to detect emotion in text. DeepEmotex is an extension of our previous work called Emotex [16]. Emotex requires extensive hand-crafted features to achieve a high performance. Such hand-crafted features are time-consuming to create and they are often incomplete. To solve this problem, we develop a deep transfer learning approach to be able to extract domain-specific features based on the context instead of using static hand-crafted features.

To avoid overfitting to our target task, DeepEmotex utilizes a large set of emotion-labeled data easily obtainable in Twitter. We study the effects of varying the amount of data used for fine-tuning the pre-trained models. We verify if increasing the amount of fine-tuning data reduces the effect of the forgetting problem in transferring knowledge from the pre-tranied language domain (i.e., BERT) to our target domain. Our evaluation results show that fine-tuning with a large set of emotion-labeled data improves both the robustness and effectiveness of the resulting target task model. Overall, we made the following contributions in this paper.

- We develop DeepEmotex to classify emotion in text messages using deep transfer learning. DeepEmotex develops transfer learning models for fine-tuning pre-trained language models and learns emotion specific features which are more contextually aware of a new domain.
- DeepEmotex utilizes two state-of-the-art pre-trained models, known as BERT [14] and USE [13]. To avoid overfitting to the target task we utilize a large set of emotion-labeled data collected from Twitter messages.
- We analyze the adaptation or fine-tuning phase during which the pretrained knowledge is transferred to our emotion classification task. DeepEmotex achieves 92% classification accuracy on our test data by fine-tuning BERT.
- We evaluate the performance of DeepEmotex models to classify emotion in EmoInt and Stimulus benchmark datasets. DeepEmotex models were able to correctly classify emotion in 73% of the instances in the benchmark datasets.
- We verify the effect of the size of data for the fine-tuning task on the accuracy of our models. Our evaluation results show that fine-tuning with a large set (more than 30K) of emotion-labeled data improves both the robustness and effectiveness of the resulting target task model.
- We evaluate DeepEmotex models by comparing their results with a baseline model (i.e., Bi-LSTM). Our evaluation results show that the proposed DeepEmotex-BERT model outperforms the baseline model by 23%.

The remainder of the paper is organized as follows: we first overview related literature on emotion classification in text using deep learning methods in Section II. A brief background knowledge in sequential transfer learning is provided in Section III. Section IV describes the details of the proposed DeepEmotex model to detect emotion using transfer learning. Section V outlines the experimental setup, and Section VI discusses the empirical results and analysis. Finally, Section VII presents the conclusion and introduces the next research direction.

II. Related Work

Recently, approaches which employ deep learning methods for emotion and sentiment detection in text have been proposed. Deep learning based methods use distributed word embeddings as input, which already encode some semantic and syntactic information.

Ren. et al. [17] proposed a context based neural network model for Twitter sentiment analysis by incorporating contextualized features from relevant Tweets into the model in the form of word embedding vectors. They showed that improvements can be achieved by modeling the context of a given target tweet using neural pooling functions to extract the most useful features from tweets. Another context-sensitive method for sentiment classification proposed by Teng et al. [18]. Their method is based on a simple weighted-sum model, using bidirectional LSTM to learn the sentiment strength, intensification and negation of lexicon sentiments in composing the sentiment value of a sentence.

Wang et al. [19] combined CNN and LSTM to capture both local information within sentences and long-distance dependency across sentences. They proposed a regional CNN-LSTM model, which consists of two parts: regional CNN and LSTM, to predict the valence and arousal ratings of text.

Due to the lack of emotion-labelled datasets, many supervised classification algorithms for emotions have been done on data gathered from microblog such as Twitter, using hashtags or emoticons as the emotion label for the data.

Felbo et al. [20] used millions of emoji occurrences in social networks as noisy labels for pre-training neural models in order to learn representations of emotional contexts. To capture the context of each word they used two bidirectional LSTM layers with 1024 hidden units (512 in each direction), with an attention layer that takes all of LSTM layers as input using skip-connections. Their results confirmed that the distant supervision to a more diverse set of noisy labels enables the models to learn better representations of emotional content in text and obtain better performance for detecting sentiment, emotions and sarcasm.

Yu et al. [21] proposed a transfer learning method using LSTM and a dual attention network. They divided the sentence representation into two feature spaces to capture the general sentiment words and emotion-specific words. A similar approach [22] utilizes a custom LSTM architecture in order to assign emotion labels to conversations in social media. They labelled their dataset through a heuristic procedure and then reconstructed this heuristic with their classifier.

Koper et al. [23] predicted emotion intensity in tweets by applying deep learning method with extended lexicons of affective norms. They showed that domain-specific embeddings (trained on twitter data) perform superior to other embeddings.

Polignano et al. [24] studied that an architecture composed of BiLSTM and self-attention demonstrated promising results in different datasets. the authors used transfer learning, using word embeddings already pre-trained in different domains. Thus, they could reduce the computational cost while covering a wider range of terms regardless of domains. The performance of their model for the ISEAR dataset is about %62.

Sent2affect proposed by Kratzwald et al. [25] is a form of transfer learning for affective computing. Their BiLSTM network is pre-trained for a different task (i.e. sentiment analysis), while the output layer is subsequently tuned to the task of emotion recognition. The resulting performance is evaluated across 6 benchmark datasets, They found BiLSTM with pre-trained word embeddings as the superior method in all experiments. We further identify that the BiLSTM appears to outperform the unidirectional LSTM in all experiments.

Batbaatar et al. [26] proposed a semantic-emotion neural network (SENN) that combines two sub-networks to capture semantic and emotional information. The first network consists of a BiLSTM to capture semantic information. The second network is a CNN to capture emotional information. The performance of the SENN model for the ISEAR dataset, is about %74. Imran et al. [27] analyzed emotion of citizens from different cultures to the Coronavirus. LSTM models used for estimating the sentiment polarity and emotions from extracted tweets have been trained to achieve state-of-the-art accuracy on the sentiment140 dataset.

Deep learning based methods mostly use distributed word vectors. Commonly used embeddings are Word2Vec [5], GloVe [6], and FastText [28]. Word2Vec and GloVe treat words as the smallest units. FastText uses a different approach where it treats each individual word as being made of n-gram characters. Thus, it can handle rare words which are not present in the dictionary.

More recently, contextualized word embeddings are proposed, called USE [13] and BERT [14], to incorporate context information in conventional word embeddings. However, these word embeddings are generalized on various tasks and limited to provide emotion information, therefore learning task-specific emotion embedding with the neural network has been proven to be effective.

Some researchers predict emotion in textual dialogues. Luo and Wang [29] fine-tune BERT model to predict emotion in dialogues by choosing from four emotion classes, joy, sadness, anger, and neutral. They use datasets consisting of scripts from the TV show, Friends, and the anonymous Facebook chat logs named EmotionPush. Chatterjee et al. [30] use BiLSTM model to infer the underlying emotion from textual dialogues by choosing from four emotion classes, happy, sad, angry, and other. EmoDet2 is another work to classify emotion in dialogues developed by Al-Omari et al. [31]. They classify EmoContext dataset into happy, sad, angry and other. They use GloVe embeddings and features extracted from AffectiveTweets. They also extract word contextual embeddings from BERT model. These vectors feed feed-forward and BiLSTM models to obtain predictions. Their result show that the performance of the system is increased by extracting BERT embeddings then feeding them into an BiLSTM network.

III. Sequential Transfer Learning

Transfer learning aims to leverage knowledge from a source task to improve the performance of a model in a different, but related target task. Sequential transfer learning learns source tasks and target tasks in sequence, and transfers knowledge from source tasks to improve the models performance on target tasks [9], [11].

Sequential transfer learning typically consists of two stages: pretraining and adaptation. During pretraining, the model is trained on source tasks. During adaptation, the pretrained model is further trained on target tasks. The standard adaptation methods includes fine-tuning and feature extraction. Fine-tuning updates parameters of the pretrained model, while feature extraction considers the pretrained model as a feature extractor and keeps the parameters fixed during the adaptation phase [20], [32].

Sequential transfer learning has been widely used recently, and deep pretrained language models have achieved great success on various NLP tasks [14], [33]. While the adaptation of the deep pretrained models is very efficient, it is prone to forgetting, where the model forgets previously learned knowledge from source tasks when learning target tasks [8].

IV. DeepEmotex: Emotion Detection using Sequential Transfer Learning

Deep pretrained language models, such as USE [13] and BERT [14], have been used in natural language processing. A wide range of NLP tasks has been promoted by these pretrained language models through Sequential Transfer Learning [8]: pretrain a language model on large-scale unlabeled data and then adapt it to downstream tasks. The adaptation step is usually conducted in two methods: fine-tuning or freezing pretrained weights. In practice, fine-tuning is adopted more widely due to its flexibility [34], [35].

DeepEmotex utilizes sequential transfer learning approach, where source and target tasks are learned in sequence. That means, models are not optimized jointly as in multi-task learning but each task is learned separately [11].

Sequential transfer learning consists of two stages: a pretraining phase in which general-purpose representations are learned on a source task or domain, and an adaptation or fine-tuning phase during which the learned knowledge is transferred to a target task or domain [11].

Using deep neural networks architectures is not trivial for emotion classification. The problem is that the pretrained models are not suited to the small emotion datasets. They typically get overfitted and suffer forgetting problem when fine-tuned with a classifier [8]. This is especially a problem when the fine-tuning data is small. To remedy the forgetting problem in transferring deep pretrained models, existing efforts mainly explore fine-tuning tricks to forget less. To fine-tune with less forgetting, we utilize a large set of emotion-labeled data easily obtainable in Twitter.

Most prior work has focused on different pretraining objectives to learn general-purpose word or sentence representations [5], [36]. A few works have explored the fine-tuning phase

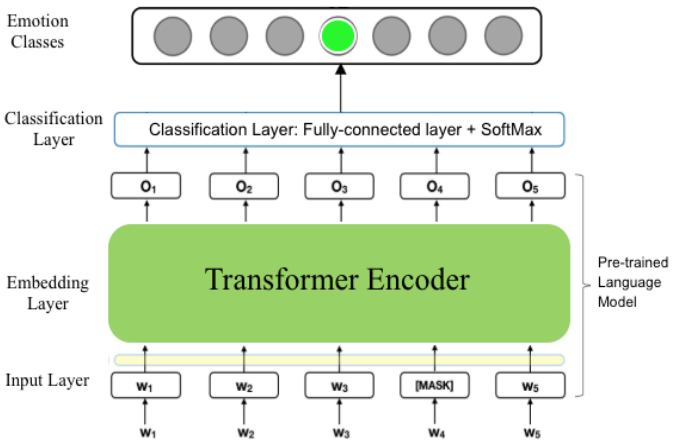

Fig. 1. Model of DeepEmotex. The embedding layer learns an embedding that contains the semantic textual information in input text. The learned representations are fed into the classification layer for emotion prediction.

and how to adapt the pretrained model to a given target task [34], [32], [15], [37]. There are two common approaches for fine-tuning: The first approach is to use the pre-training network as a feature extractor [38], where all layers in the model are frozen when fine-tuning on the target task except the last layer. In this approach the pre-trained representations are used in a downstream model. Alternatively, the pre-trained model's parameters are unfrozen and fine-tuned on a new task [39]. This approach enables to adapt a general-purpose representation to many different tasks.

Figure 1 shows our DeepEmotex model. Our model represents input words by their embeddings. Following the embedding layer, our model consists of a transformer encoder, followed by a SoftMax classification layer. Gaining a better understanding of the adaptation phase is key in making the most use out of pre-trained representations. Accordingly, DeepEmotex utilizes two state-of-the-art pre-trained models, known as BERT [14] and Universal Sentence Encoder [13]. Using these models, we transfer knowledge learned from a large corpus to our emotion classification model. We then fine-tune DeepEmotex model to fit to our target emotion datasets.

A. DeepEmotex-USE: A Transfer Learning Model using Universal Sentence Encoder

Universal Sentence Encoder (USE) [13] is a deep neural network to create universal sentence embeddings. Universal embeddings are pretrained embeddings obtained from training deep learning models on a huge corpus. These pretrained (generic) embeddings can be used in a wide variety of NLP tasks including text classification, semantic similarity and clustering.

The Universal Sentence Encoder is trained and optimized for greater-than-word length text, such as sentences, phrases or short paragraphs. The input is a variable length text. The Universal Sentence Encoder encodes the input text into 512-

dimensional embeddings. The USE embeddings are trained on different data sources and tasks with the aim of dynamically accommodating a wide variety of natural language understanding tasks which require modeling the meaning of word sequences rather than just individual words.

Essentially, there are two versions of the USE models. The first version makes use of a Deep Averaging Network (DAN) where input embeddings for words and bi-grams are first averaged together and then passed through a feed-forward deep neural network (DNN) to produce sentence embeddings [13]. Deep Averaging Network (DAN) is simpler than the second version. The primary advantage of the DAN encoder is that its compute time is linear in the length of the input sequence.

The second version makes use of the transformer-network based sentence encoding model. The transformer encoder is composed of a stack of N = 6 identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-wise fully connected feed-forward network, followed by layer normalization.

Their results [13] demonstrate that the transformer-based encoder achieves the best overall transfer task performance. However, this comes at the cost of computing time and memory usage scaling dramatically with sentence length. For our emotion classification task we use the transformer-based encoder as it achieves better overall performance than the DAN encoder.

We apply transfer learning leveraging prior knowledge from pre-trained embeddings to solve our emotion classification task. Our sentence embedding sub-network leverages the Universal Sentence Encoder. We fine-tune the sentence embeddings using our collected emotion-labeled dataset.

We build a feed-forward neural network with two hidden dense layers and the rectified linear activation function (ReLU). ReLU overcomes the vanishing gradient problem. This is good for deep neural networks which suffer from the vanishing and explosion gradient problem [40].

A dense layer provides learning features from all the combinations of the features of the previous layer. The input of our model is 512-feature vectors created using the Universal Sentence Encoder technology. The resulting vector is then fed into fully connected layers culminating in a softmax layer. We then fine-tune our model using collected labeled tweets introduced in Section V-A. We fine-tune the embedding weights by setting the trainable parameter to true. Here we leverage transfer learning in the form of pre-trained embeddings.

The overall architecture of DeepEmotex-USE is shown in Figure 2. The input of the model is a twitter message. First, the embedding layer uses the pre-trained USE model to map a sentence into its embedding vector. The model that we are using splits the sentence into tokens, embeds each token and then combines them into context-aware 512-dimension embeddings. Then, the embeddings are passed through a feed-forward neural network with ReLU activation. It projects the input into 256-dimension embeddings and feeds them to the classification layer to produce a classification probability. The output of our model is an emotion classification label. The main objective is to correctly predict the emotion of each tweet.

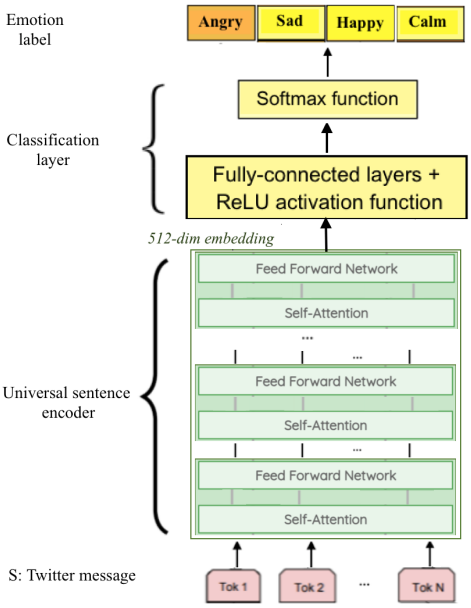

Fig. 2. Model of DeepEmotex-USE to classify emotion in text messages using Universal Sentence Encoder.

B. DeepEmotex-BERT: A Transfer Learning Model using Bidirectional Encoder Representations from Transformers

The model architecture of DeepEmotex-BERT using Bidirectional Encoder Representations from Transformers (BERT) is shown in Figure 3. The input of the model is a twitter message and the output is an emotion label. We use the pre-trained BERT model [14] to generate text representations. BERT learns text representations using a bidirectional Transformer encoder [7] pre-trained on the language modeling task. Transformers have a sequence-to-sequence model architecture. Each transformer includes a separate encoder and decoder component. The difference is in their use of attention known as self-attention. The core architecture consists of a stack of encoders fully connected to a stack of decoders. Each encoder consists of a self-attention component and a feed forward network. Each decoder consists of a self-attention component, an encoder-decoder attention component, and a feed forward component.

BERT has several variants based on model configurations. We adopt BERT-base [7] as our base model. BERT-base consists of an encoder with 12-layer Transformer blocks. For each block in the encoder, it contains a 12-head self-attention layer and 768-dimensional hidden layer, yielding a total of 110M parameters. The base model allows inputs up to a sequence of 512 tokens and outputs the vector representations of the sequence. The input sequence has one or two segments. The first token of the sequence is always $[CLS]$ which contains the special classification embedding. Another special token

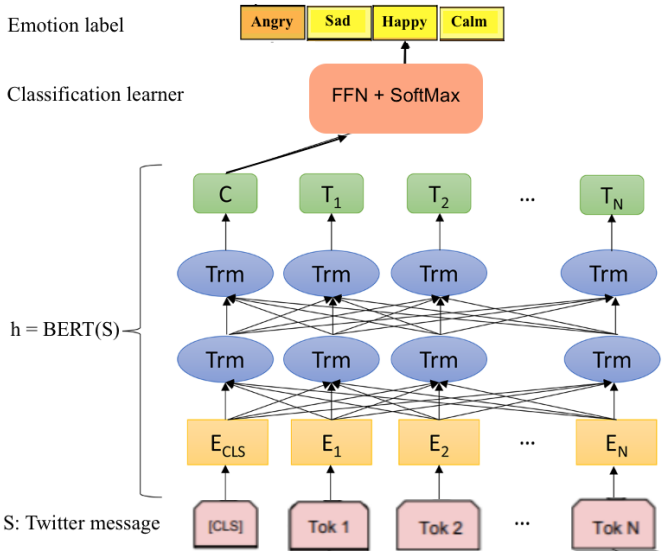

Fig. 3. Model of DeepEmotex to classify emotion in text messages using pretrained BERT.

$[SEP]$ is used for separating segments. In order to facilitate the training and fine-tuning of BERT model, we transform the input text into $[CLS] + text + [SEP]$ format.

We follow Devlin et al. [14] and create a fully connected layer over the final hidden state corresponding to the $[CLS]$ input token. During fine-tuning, we optimize the entire model, with the additional softmax classifier parameters $W \in R^{K \times H}$, where H is the dimension of the hidden state vectors and $K = 4$ is the number of emotion classes.

Let $S = ([CLS], t_1, \ldots, t_m, [SEP], ..., t_n)$ be the input sequence (i.e., Twitter message), where $t_1, \ldots, t_m$ denotes a sentence with m tokens. For emotion classification task, we use the final hidden state $h = BERT(S)$ of the first token $[CLS]$ as the representation of the whole sequence S. A standard softmax classifier added on top of BERT predicts the probability of emotion label c:

$$x = W \cdot h + b$$

$$P(c|x) = softmax(x) = \frac{\exp(x)}{\sum_{k=1}^{C} \exp(x)}$$

where W is the task-specific weight matrix, and b is the bias vector to be estimated. We fine-tune all parameters as well as W jointly by maximizing the log-probability of the correct label.

V. DeepEmotex: Experimental Methodology

After developing DeepEmotex models, we run several experiments to fine-tune pretranied models based on the emotion-classification task using our input dataset. We conduct the experiments with the deep pretrained language model BERT-base [14] and USE [13].

TABLE II
Number of tweets collected as emotion-labeled data

Class	Happy-Active	Happy-Inactive	Unhappy-Active	Unhappy-Inactive	Total
#Tweets	148,571	195,313	149,287	47,354	540,525

A. DeepEmotex Emotion Dataset

Deep pretrained models get overfitted to small datasets and suffer forgetting problem when fine-tuned with a classifier [8]. To fine-tune with less forgetting, we utilize a large set of emotion-labeled data obtainable in Twitter.

In order to fine-tune pretranied models for our emotion classification task, we need a large dataset of labeled tweets. Since there are not many human-labeled datasets publicly available, we collect tweets with emotion-carrying hashtags as a surrogate for emotion labels [41], [16], [42]. We define four emotion classes (i.e., joy, relax, anger, and sadness) based on the emotion model proposed by Circumplex model [43]. We only collect the tweets labeled with these emotions.

In order to collect enough tweets to serve as our labeled dataset, we develop a list of hashtags representing each of four emotion classes proposed by the Circumplex model. For each emotion class, we define a set of hashtags representing the emotion. We then use the set of hashtags to collect tweets with the emotion hashtags at the end.

One advantage of using hashtags for labeling emotion data is that the label is assigned by the writer of the tweet rather than an annotator who could wrongly decide the category of a tweet. After all, emotion is a fuzzy concept. Another advantage of this method is obviously that it enables us to acquire a sufficiently large labeled dataset to apply fine-tuning in our transfer learning approach.

Twitter data is very noisy, not only because of use of non-standard typography, but due to many duplicate tweets and the fact that tweets often have multiple emotion hashtags. Since these reduce our ability to build accurate models, we clean the data. We first apply some general data cleaning strategies. We convert the tweets to lower-case letters. We remove non-ascii letters, urls, "@name" and duplicate letters. We also filter out all retweets based on existence of the token "RT" regardless of its case. Since our goal is to create non-overlapping classes at the level of a tweet, we also remove all tweets with hashtags belonging to more than one emotion of the four emotion classes.

Table II represents the number of labeled tweets selected for each class after pre-processing. The labeled data are relatively balanced. The number of data labeled relax is especially less than other classes. Our dataset comprises a total of 540,525 tweets labeled with four emotion classes. We shuffle our labeled datasets and create train, validation and test datasets. We train our models on a total of 300,000 tweets as our training dataset, validate on 60,000 tweets. We use 180,525 tweets as our test dataset.

B. Experimental Setup of the DeepEmotex-USE Model

Universal Sentence Encoder model can be fine-tuned to our target task in several ways by freezing layers to disable parameters updates. One common approach is to use the network as a feature extractor [38], where all layers in the model are frozen during fine-tuning on the target task except the last layer. Another common approach is to use the pre-trained model as an initialization, and thereafter the full model is unfrozen [20]. We implement both approaches to fine-tune Universal Sentence Encoder. Our first transfer learning approach is implemented using the following workflow:

- Instantiate a base model and load pre-trained weights into it.
- Freeze all layers in the base model by setting trainable = False.
- Add classification layer (Fully connected layer + Soft-Max) on top of the frozen layers.
- Train the new layers on our new dataset.

Next, we implement the second approach. For this, we unfreeze all layers and re-train the whole model on our dataset for several epochs. This helps fine-tune the model towards our task by incrementally adapting the pre-trained features to our new data. We fine-tune the embedding weights by setting trainable=true. Weights are updated (via gradient descent) to minimize the loss during training. By training all layers of the model we are able to adjust the parameters across the network during back propagation. Our second transfer learning approach is implemented using the following workflow:

- Instantiate a base model and load pre-trained weights into it.
- Unfreeze all layers in the base model by setting trainable = True.
- Add classification layer (Fully connected layer + Soft-Max) on top of the USE unfrozen layers.
- Train all the layers on our new dataset to fine-tune the old parameters on the new dataset.

We use the Universal Sentence Encoder Version 3 [1] as our base model. We add a feed-forward neural network with two hidden layers and the Relu activation function.

We set our batch size to 150 and number of epochs to 20. Batch-size defines the number of examples will be passed to our model during one iteration, and number of epochs is the number of times our model will go through the entire training set. We train the model on our training datasets using Adam optimizer with a learning rate of 0.001. The performance of the re-trained model is evaluated at the end of training epochs with our test datasets.

C. Experimental Setup of the DeepEmotex-BERT Model

BERT uses books corpus and Wikipedia data sources to pre-train their models. Although BERT aims to learn contextualized representations across a wide range of NLP tasks, leveraging BERT alone still leaves the domain challenges unresolved as BERT is only trained on formal texts and has almost no understanding of social media text [44]. The end tasks from the original BERT paper typically use tens of thousands of examples to ensure that the system is task-aware [44]. Below, we introduce fine-tuning BERT to boost the performance of classifying emotion on Twitter messages.

We adopt the pre-trained BERT-BASE as the encoding layer of our DeepEmotex model. We extend them with extra tasks-specific layers and fine-tune the model on our emotion classification task. We focus on fine-tuning a classification layer implemented as a standard feed-forward and a softmax layer on top of the pre-trained BERT.

For fine-tuning the target model, we keep the hyper-parameters the same as in the pre-training by Devlin [14], except for the batch size, learning rate, and number of training epochs. Devlin et al. found the following range of possible values to work well across different tasks [14]:

- Batch size: 32, 64
- Learning rate: 5e-5, 3e-5, 2e-5
- Number of epochs: 2, 3, 4

The optimal hyper-parameter values are task-specific. For our model, we decide to choose a small learning rate, and train with a few epochs. Most of transfer learning models suffer from the so-called catastrophic forgetting problem. That is, the learnt knowledge is diminished when learning the new information from the target domain. Using a large learning rate such as 6e-4 makes the training fail to converge. The BERT authors [14] recommend a number of training epochs between 2 and 4. Selecting a large number of epochs may cause over-fitting. As shown in Table III, we fine-tune our model for 2 epochs with a learning rate of 4e-5. A larger batch size often results in lower accuracy but faster epochs. In order to find the optimum batch size we perform several runs of varying batch sizes while keeping other parameters constant.

VI. Experimental Results of DeepEmotex mdels

In this section, we evaluate our approach and report empirical results.

A. Experimental Results of DeepEmotex-USE

We train our USE models (with frozen and unfrozen layers) for 20 epochs. The training is conducted on our collected tweets. Our dataset comprises a total of 540,525 tweets labeled with four emotion classes as described in Section V-A. We first shuffle our labeled datasets and create train, validation and test datasets. The training data is used for fine-tuning and the testing dataset is used for evaluation.

Figure 4 shows the classification accuracy of our two models with frozen and unfrozen layers on the validation set in terms of micro-average F1 score. As it shows, the accuracy of our models stabilize at about epoch 14. The final validation results show that our model with frozen layers gets an accuracy of $90\% \pm 0.82\%$ (Mean $\pm$ Standard deviation) after training for 20 epochs. The frozen model gets an accuracy of 90.6% on our test dataset. This is consistent with our validation dataset.

[1] https://tfhub.dev/google/universal-sentence-encoder-large/3

TABLE III
PARAMETERS OF FINE-TUNING BERT

Parameters	Epoch	Batch sizes	Learning rate
Values	2	50,100,150,200,250	4e-5

The final validation results show that our model with unfrozen layers gets an accuracy of $90.6\% \pm 0.85\%$ (Mean $\pm$ Standard deviation) after training for 20 epochs. We get an accuracy of 91% on our test dataset using the model with unfrozen layers. Comparing the results of the two models with frozen and unfrozen layers using statistical t-test shows that the unfreezing approach performed better than the freezing approach ($p - value = 0.007$).

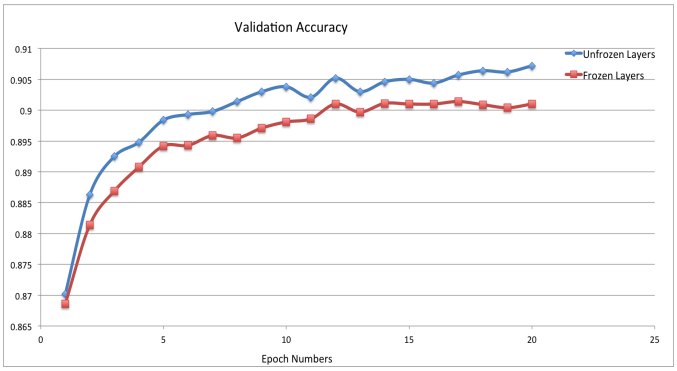

Fig. 4. Emotion Classification Accuracy of DeepEmotex-USE with frozen and unfrozen layers on validation set in terms of micro-average F1 score.

B. Experimental Results of DeepEmotex-BERT

Fine-tuning our BERT model is conducted on collected tweets. Our dataset comprises a total of 540,525 tweets labeled with four emotion classes. The collected tweets are shuffled and divided into training, validation and test datasets as described in Section V-A.

We fine-tune our BERT model using different batch sizes for 2 epochs. Figure 5 shows the classification performance of DeepEmotex models created using different batch sizes to fine-tune BERT. As it can be seen in Figure 5 the model achieved a performance of $91.8\% \pm 0.2\%$ on our test data in terms of F1 score. The highest performance 92% is achieved when the batch size is 100.

C. Evaluating DeepEmotex Models

After having fine-tuned the models of DeepEmotex, we now set out to evaluate its generality to classify emotion. For this, we evaluate the performance of the DeepEmotex models to classify emotion in popular benchmark data sets. An issue with many of the benchmark datasets is data scarcity, which is specially problematic in emotion analysis [20].

To evaluate our DeepEmotex method on emotion detection we make use of EmoInt [45] and Stimulus [46] as benchmark datasets. EmoInt (Emotion Intensity in Tweets) is a dataset of emotions in tweets from WASSA 2017 [45]. The benchmark

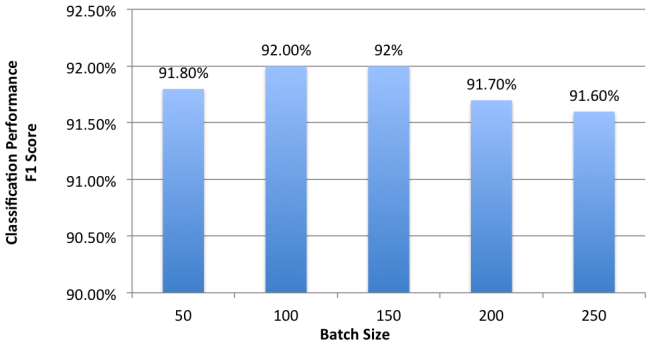

Fig. 5. Classification results of DeepEmotex-BERT on our test dataset using different batch sizes in terms of micro-average F1 score.

TABLE IV
NUMBER OF MESSAGES IN EACH EMOTION CLASS IN OUR BENCHMARK DATASETS

Emotion Dataset	Joy	Sadness	Anger
Stimulus Dataset	480	540	570
EmoInt Dataset	671	642	656

dataset provides four emotion classes including joy, anger, fear, and sadness. Emotion Stimulus dataset [46] contains 820 sentences annotated with both an emotion cause and an emotion tag. Their taxonomy is joy, anger, sadness, disgust, fear, shame and surprise.

We evaluate three emotion classes, including joy, anger, and sadness. Table IV lists the number of samples in each class in our benchmark datasets. As can be seen in Table IV, the benchmark data sets are balanced, that is, there are approximately an equal number of examples that belong to each of the four classes.

Our fine-tuned models are utilized to classify the benchmark dataset. These models classify the input benchmark dataset into our four emotion classes. The input class joy is correctly classified if it is labeled as happy-active or happy-inactive. The input class anger is correctly classified if it is labeled as unhappy-active, and the input class sadness is correctly classified if it is labeled as unhappy-inactive.

To further evaluate the predictive performance of DeepEmotex models, we compare the classification results of our proposed models against the results of a baseline model. We train a bidirectional LSTM as a baseline model, which have shown success in various other domains.

Tables V and VI list the evaluation results of the proposed DeepEmotex models to classify emotion in EmoInt dataset [45] and Stimulus dataset [46] in terms of F-score. The first Three columns include the result of DeepEmotex-BERT and DeepEmotex-USE with freezing and unfreezing layers. The last column of the tables lists the results of the baseline model. As table VI and VI presents, The DeepEmotex-BERT model outperforms other models. The DeepEmotex-USE model created with unfreezing layers achieves slightly higher classification accuracy than the model created with

TABLE V
CLASSIFICATION ACCURACY OF DIFFERENT MODELS TO PREDICT EMOTION IN EMOINT BENCHMARK DATASET IN TERMS OF F-SCORE

Model / Emotion Class	DeepEmotex-BERT	DeepEmotex-USE UnFrozen layers	DeepEmotex-USE Frozen layers	Bi-LSTM
Joy	**78.4%**	60%	66%	60%
Anger	**57.3%**	56%	53.3%	33%
Sadness	**71.4%**	51.3%	50.4%	47%
Micro Average F-score	**71%**	58%	57 %	50%

TABLE VI
CLASSIFICATION ACCURACY OF DIFFERENT MODELS TO PREDICT EMOTION IN STIMULUS BENCHMARK DATASET IN TERMS OF F-SCORE

Model / Emotion Class	DeepEmotex-BERT	DeepEmotex-USE UnFrozen layers	DeepEmotex-USE Frozen layers	Bi-LSTM
Joy	**63%**	57%	60%	50.3%
Anger	**77%**	62%	60%	35%
Sadness	**77%**	61%	58.3%	56%
Micro Average F-score	**73.5%**	60	59 %	50%

freezing layers.

Comparing the evaluation results of Table V with Table VI shows that the DeepEmotex-BERT models achieve a higher performance than DeepEmotex-USE models in classifying emotion in both benchmark datasets. These evaluation results show that the proposed DeepEmotex-BERT and DeepEmotex-USE models outperform the baseline model (BiLSTM). Overall, our DeepEmotex-BERT model achieves a higher accuracy than the baseline model (i.e., BiLSTM) by 21% on EmoInt dataset and by 23.5% on Stimulus dataset.

We also investigate if the size of data during the fine-tuning task has an influence on the performance of our models. We verify whether using more data would diminish the catastrophic forgetting problem and if it would be worthwhile gathering more data for fine-tuning the pretrained models. Figure 6 presents the classification results of the DeepEmotex-BERT model using different size of fine-tuning dataset to predict emotion in the benchmark datasets. As Figure 6 shows, increasing the amount of data, with which our DeepEmotex-BERT model was fine-tuned from 3K to 30K, improves the performance of the model to classify emotion by 18% in the Stimulus dataset and by 9% in the EmoInt dataset. Increasing the amount of fine-tuning data from 30K to 300K slightly increased the classification performance of our DeepEmotex-BERT model. The trend suggests that the proposed model is expressive enough to learn from more data. Moreover, the proposed model reduces the effect of the forgetting problem in transferring knowledge from the pre-tranied language domain (i.e., BERT) to our target domain.

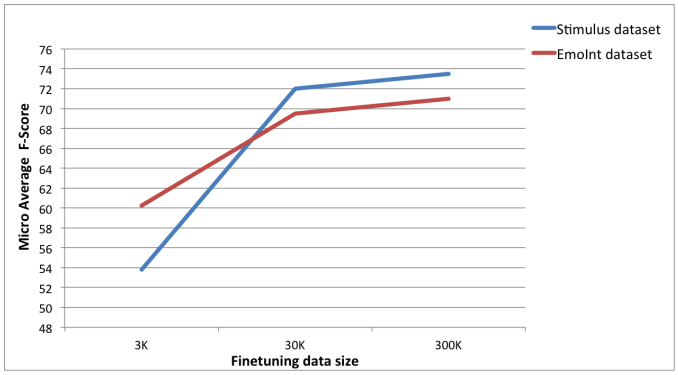

Fig. 6. Effect of the size of the fine-tuning dataset on classification performance of DeepEmotx-BERT model to predict emotion of the benchmark datasets in terms of Micro Average F-score.

VII. CONCLUSION

In this paper, we study the effectiveness of deep transfer learning methods to detect emotion in text messages. We develop and evaluate DeepEmotex models to automatically classify emotion in text messages. DeepEmotex models learn feature representations for emotion classification using sequential transfer learning. More precisely, DeepEmotex uses USE [13] and BERT [14] as pre-trained models. These models are then fine-tuned on our emotion classification task and obtain state-of-the-art results.

Our experimental results demonstrate the effectiveness and robustness of the fine-tuned DeepEmotex models. Our DeepEmotex models were able to achieve over 91% accuracy for multi-class emotion classification on test dataset.

We evaluated the performance of the models created using BERT and the models created using USE in classifying emotion in the EmoInt benchmark dataset. Our DeepEmotex-BERT models were able to correctly classify emotion in 73% of the instances in the benchmark dataset. Our evaluation results show that the proposed DeepEmotex-BERT models outperform the baseline Bi-LSTM model [47] by 23%.

We also studied the effect of varying the amount of data used for fine-tuning the pretrained models. We observed that using more data for fine-tuning the pre-trained model improved the classification performance of the target task by 20% in the Stimulus dataset and by 11% in the EmoInt dataset.

REFERENCES

[1] M. Hasan, E. Rundensteiner, and E. Agu, "Automatic emotion detection in text streams by analyzing twitter data," *International Journal of Data Science and Analytics*, vol. 7, no. 1, pp. 35–51, 2019.

[2] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[3] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[4] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *ieee Computational intelligenCe magazine*, vol. 13, no. 3, pp. 55–75, 2018.

[5] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[6] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[8] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," *arXiv preprint arXiv:1801.06146*, 2018.

[9] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020.

[10] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," *arXiv preprint arXiv:1902.00751*, 2019.

[11] S. Ruder, M. E. Peters, S. Swayamdipta, and T. Wolf, "Transfer learning in natural language processing," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, 2019, pp. 15–18.

[12] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1, no. 2.

[13] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar *et al.*, "Universal sentence encoder," *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

[14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, 2018.

[15] J. Phang, T. Févry, and S. R. Bowman, "Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks," *arXiv preprint arXiv:1811.01088*, 2018.

[16] M. Hasan, E. Rundensteiner, and E. Agu, "Emotex: Detecting emotions in twitter messages," in *Proceedings of the Sixth ASE International Conference on Social Computing (SocialCom 2014)*. Academy of Science and Engineering (ASE), USA, 2014.

[17] Y. Ren, Y. Zhang, M. Zhang, and D. Ji, "Context-sensitive twitter sentiment classification using neural network." in *AAAI*, 2016, pp. 215–221.

[18] Z. Teng, D. T. Vo, and Y. Zhang, "Context-sensitive lexicon features for neural sentiment analysis," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016, pp. 1629–1638.

[19] J. Wang, L.-C. Yu, K. R. Lai, and X. Zhang, "Dimensional sentiment analysis using a regional cnn-lstm model," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, vol. 2, 2016, pp. 225–230.

[20] B. Felbo, A. Mislove, A. Søgaard, I. Rahwan, and S. Lehmann, "Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 1615–1625.

[21] J. Yu, L. Marujo, J. Jiang, P. Karuturi, and W. Brendel, "Improving multi-label emotion classification via sentiment classification with dual attention transfer network." ACL, 2018.

[22] U. Gupta, A. Chatterjee, R. Srikanth, and P. Agrawal, "A sentiment-and-semantics-based approach for emotion detection in textual conversations," *arXiv preprint arXiv:1707.06996*, 2017.

[23] M. Köper, E. Kim, and R. Klinger, "Ims at emoint-2017: Emotion intensity prediction with affective norms, automatically extended resources and deep learning," in *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 2017, pp. 50–57.

[24] M. Polignano, P. Basile, M. de Gemmis, and G. Semeraro, "A comparison of word-embeddings in emotion detection from text using bilstm, cnn and self-attention," in *Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization*, 2019, pp. 63–68.

[25] B. Kratzwald, S. Ilić, M. Kraus, S. Feuerriegel, and H. Prendinger, "Deep learning for affective computing: Text-based emotion recognition in decision support," *Decision Support Systems*, vol. 115, pp. 24–35, 2018.

[26] E. Batbaatar, M. Li, and K. H. Ryu, "Semantic-emotion neural network for emotion recognition from text," *IEEE Access*, vol. 7, pp. 111 866–111 878, 2019.

[27] A. S. Imran, S. M. Daudpota, Z. Kastrati, and R. Batra, "Cross-cultural polarity and emotion detection using sentiment analysis and deep learning on covid-19 related tweets," *IEEE Access*, vol. 8, pp. 181 074–181 090, 2020.

[28] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[29] L. Luo and Y. Wang, "Emotionx-hsu: Adopting pre-trained bert for emotion classification," *arXiv preprint arXiv:1907.09669*, 2019.

[30] A. Chatterjee, K. N. Narahari, M. Joshi, and P. Agrawal, "Semeval-2019 task 3: Emocontext contextual emotion detection in text," in *Proceedings of the 13th International Workshop on Semantic Evaluation*, 2019, pp. 39–48.

[31] H. Al-Omari, M. A. Abdullah, and S. Shaikh, "Emodet2: Emotion detection in english textual dialogue using bert and bilstm models," in *2020 11th International Conference on Information and Communication Systems (ICICS)*. IEEE, 2020, pp. 226–232.

[32] S. Chen, Y. Hou, Y. Cui, W. Che, T. Liu, and X. Yu, "Recall and learn: Fine-tuning deep pretrained language models with less forgetting," *arXiv preprint arXiv:2004.12651*, 2020.

[33] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[34] M. Peters, S. Ruder, and N. A. Smith, "To tune or not to tune? adapting pretrained representations to diverse tasks," pp. 7–14, 2019.

[35] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "ALBERT: A lite BERT for self-supervised learning of language representations," *CoRR*, vol. abs/1909.11942, 2019.

[36] R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler, "Skip-thought vectors," in *Advances in neural information processing systems*, 2015, pp. 3294–3302.

[37] W. Ragheb, J. Azé, S. Bringay, and M. Servajean, "Attention-based modeling for emotion detection and classification in textual conversations," *arXiv preprint arXiv:1906.07020*, 2019.

[38] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *International conference on machine learning*, 2014, pp. 647–655.

[39] A. M. Dai and Q. V. Le, "Semi-supervised sequence learning," in *Advances in neural information processing systems*, 2015, pp. 3079–3087.

[40] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, p. e1253, 2018.

[41] M. Hasan, E. Agu, and E. Rundensteiner, "Using hashtags as labels for supervised learning of emotions in twitter messages," in *Proceedings of the ACM SIGKDD Workshop on Healthcare Informatics, HI-KDD*, 2014.

[42] S. Mohammad, F. Bravo-Marquez, M. Salameh, and S. Kiritchenko, "Semeval-2018 task 1: Affect in tweets," in *Proceedings of the 12th international workshop on semantic evaluation*, 2018, pp. 1–17.

[43] J. A. Russell, "A circumplex model of affect," *Journal of Personality and Social Psychology*, vol. 39, pp. 1161–1178, 1980.

[44] H. Xu, B. Liu, L. Shu, and P. S. Yu, "Bert post-training for review reading comprehension and aspect-based sentiment analysis," *arXiv preprint arXiv:1904.02232*, 2019.

[45] S. M. Mohammad and F. Bravo-Marquez, "WASSA-2017 shared task on emotion intensity," in *Proceedings of the Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA)*, Copenhagen, Denmark, 2017.

[46] D. Ghazi, D. Inkpen, and S. Szpakowicz, "Detecting emotion stimuli in emotion-bearing sentences," in *International Conference on Intelligent Text Processing and Computational Linguistics*. Springer, 2015, pp. 152–165.

[47] Y. Zhang, H. Yuan, J. Wang, and X. Zhang, "Ynu-hpcc at emoint-2017: using a cnn-lstm model for sentiment intensity prediction," in *Proceedings of the 8th workshop on computational approaches to subjectivity, sentiment and social media analysis*, 2017, pp. 200–204.