Data Mining and Knowledge Discovery (2023) 37:318–380
https://doi.org/10.1007/s10618-022-00853-0

Sentiment analysis in tweets: an assessment study from classical to modern word representation models

Sérgio Barreto[1] · Ricardo Moura[1] · Jonnathan Carvalho[2] · Aline Paes[1] · Alexandre Plastino[1]

Received: 21 May 2021 / Accepted: 29 June 2022 / Published online: 15 November 2022

Abstract

With the exponential growth of social media networks, such as Twitter, plenty of user-generated data emerge daily. The short texts published on Twitter – the tweets – have earned significant attention as a rich source of information to guide many decision-making processes. However, their inherent characteristics, such as the informal, and noisy linguistic style, remain challenging to many natural language processing (NLP) tasks, including sentiment analysis. Sentiment classification is tackled mainly by machine learning-based classifiers. The literature has adopted different types of word representation models to transform tweets to vector-based inputs to feed sentiment classifiers. The representations come from simple count-based methods, such as bag-of-words, to more sophisticated ones, such as BERTweet, built upon the trendy BERT architecture. Nevertheless, most studies mainly focus on evaluating those models using only a small number of datasets. Despite the progress made in recent years in language modeling, there is still a gap regarding a robust evaluation of induced embeddings applied to sentiment analysis on tweets. Furthermore, while fine-tuning the model from downstream tasks is prominent nowadays, less attention has been given

Responsible editor: Johannes Fürnkranz.

✉ Sérgio Barreto
sergiobarreto@id.uff.br

Ricardo Moura
mouraricardo@id.uff.br

Jonnathan Carvalho
joncarv@iff.edu.br

Aline Paes
alinepaes@ic.uff.br

Alexandre Plastino
plastino@ic.uff.br

1 Universidade Federal Fluminense, Niterói, Brazil

2 Instituto Federal Fluminense, Itaperuna, Brazil

to adjustments based on the specific linguistic style of the data. In this context, this study fulfills an assessment of existing neural language models in distinguishing the sentiment expressed in tweets, by using a rich collection of 22 datasets from distinct domains and five classification algorithms. The evaluation includes static and contextualized representations. Contexts are assembled from Transformer-based autoencoder models that are also adapted based on the masked language model task, using a plethora of strategies.

Keywords Sentiment analysis · Text representations · Language models · Natural language processing · Twitter

1 Introduction

In recent years, the use of social media networks, such as Twitter[1], has been growing exponentially. It is estimated that about 500 million tweets – the short informal messages sent by Twitter users – are published daily.[2] Unlike others text style, tweets have an informal linguistic style, misspelled words, the careless use of grammar, URL links, user mentions, hashtags, and more. Due to these inherent characteristics, discovering patterns from tweets represents a challenge and opportunities for machine learning and natural language processing (NLP) tasks, such as sentiment analysis.

Sentiment analysis is the field of study that analyzes people's opinions, sentiments, appraisals, attitudes, and emotions toward entities and their attributes expressed in written text (Liu 2020). Usually, one reduces the sentiment analysis task to find out the polarity classification, i.e., whether they carry a positive or negative connotation. One of the biggest challenges concerning the sentiment classification of tweets is that people often express their sentiments and opinions using a casual linguistic style, resulting in the presence of misspelling words and the careless use of grammar. Consequently, the automated analysis of tweets' content requires machines to build a deep understanding of natural text to deal effectively with its informal structure (Pathak et al. 2020). However, before discovering patterns from text, it is essential to define a more fundamental step: how automatic methods can numerically represent textual content.

Vector space models (VSMs) (Salton et al. 1975) are one of the earliest and most common strategies adopted in text classification literature to allow for machines to deal with texts and their structures. The VSM represents each document in a corpus as a point in a vector space. Points that are close together in this space are semantically similar, and points that are far apart are semantically distant (Turney and Pantel 2010). The firsts VSM approaches are count-based methods, such as Bag-of-Words (BoW) and BoW with TF-IDF (Term Frequency-Inverse Document Frequency) (Manning et al. 2008). Although VSMs have been extensively used in the literature, they cannot deal with the curse of dimensionality. More clearly, considering the inherent characteristics of tweets, a corpus of tweets may contain different spellings for each unique

[1] http://www.twitter.com.

[2] https://www.dsayce.com/social-media/tweets-day/.

word leading to an extensive vocabulary, making the vector representation of those tweets very large and often sparse.

To tackle the curse of dimensionality inherent from BOW-based approaches, in the last years it has become a standard practice to learn dense vectors to represent words and texts, the *embeddings*. Methods such as such as Word2Vec (Mikolov et al. 2013), FastText (Mikolov et al. 2018), and others (Agrawal et al. 2018; Felbo et al. 2017; Tang et al. 2014; Xu et al. 2018) have been used with relative success to address a plethora of NLP tasks. Nevertheless, in general, the performance of such techniques are still unsatisfactory to solve sentiment analysis from tweets, taking into account the dynamic vocabulary used by Twitter users to express themselves. Specifically, in tweets, the ironic and sarcastic content expressed in a limited space, regularly out of context and informal, makes even more challenging to retrieve meaning from the words. Such attributes may degrade the performance of traditional word embeddings methods if not handled properly. In this context, contextualized word representations have recently emerged in the literature, aiming at allowing the vector representation of words to adapt to the context they appear. Contextual embedding techniques, including ELMo (Peters et al. 2018) and Transformer-based autoencoder methods, such as BERT (Devlin et al. 2019), RoBERTa (Liu et al. 2019), and BERTweet (Nguyen et al. 2020), are built upon the concept of neural language model (Bengio et al. 2000) to capture not only complex characteristics of word usage, such as syntax and semantics, but also how the word usage vary across linguistic contexts. Those methods have achieved state-of-the-art results on various NLP tasks, including sentiment analysis (Adhikari et al. 2019; Akkalyoncu Yilmaz et al. 2019; Chaybouti et al. 2021; Gao et al. 2019).

Much effort in recent language modeling research is focused on scalability issues of existing word embedding methods. On this basis, inductive transfer learning strategies and pre-trained embedding models have gained important application in the literature, especially when the amount of labeled data to train a model is relatively small. With that, models obtained from the aforementioned contextual embeddings methods are rarely trained from scratch but are instead fine-tuned from models pre-trained on datasets with a huge amount of texts (Howard and Ruder 2018; Peters et al. 2018; Radford et al. 2018). Pre-trained models reduce the use of computational resources and tend to increase the classification performance of several NLP tasks, sentiment analysis included.

Despite the successful achievements in developing efficient word representation methods in NLP literature, there is still a gap regarding a robust evaluation of existing language models applied to the sentiment analysis task on tweets. Most studies are mainly focused on evaluating those models for different NLP tasks using only a small number of datasets(Lan et al. 2020; Liu et al. 2019; Peters et al. 2018; Xu et al. 2018). Our main goal is to identify appropriate embedding-based text representations for the sentiment analysis of English tweets in this study. For this purpose, we evaluate distinct types of embeddings, including: i) static embeddings learned from generic texts (Agrawal et al. 2018; Mikolov et al. 2018, 2013; Pennington et al. 2014); ii) static embeddings learned from datasets of Twitter sentiment analysis (Araque et al. 2017; Bravo-Marquez et al. 2016; Felbo et al. 2017; Pennington et al. 2014; Tang et al. 2014; Xu et al. 2018); iii) contextualized embeddings learned from transformer-based autoencoders with generic texts with no adjustments (Devlin et al. 2019; Liu et al.

2019); iv) contextualized embeddings learned from Transformer-based autoencoders with a dataset of tweets with no adjustments (Nguyen et al. 2020); v) contextualized embeddings adapted to the tweets language with a second phase of pretraining the language model (Gururangan et al. 2020); and vi) contextualized embeddings adapted to the tweets *sentiment* language with a second phase of pretraining the language model (Gururangan et al. 2020). In all assessments, we use a representative set of twenty-two sentiment datasets (Carvalho and Plastino 2021) as input to five classifiers to evaluate the predictive performance of the embeddings. To the best of our knowledge, there is no previous study that has conducted such a robust evaluation regarding language models of several flavors and a large number of datasets. In order to identify the most appropriate text embeddings, we conduct this study to answer the following four research questions.

RQ1 Which static embeddings are the most effective in the sentiment classification of tweets? Our motivation to evaluate those models is that many state-of-the-art deep learning models can require a lot of computational power, such as memory usage and storage. Thus, running those models locally on some devices may be difficult for mass-market applications that depend on low-cost hardware. To overcome this limitation, embeddings generated by language models can be gathered by simply looking up at the embedding table to achieve a static representation of textual content. We intend to assess how these static representations work and which are the most appropriate in this context. We answer this research question by evaluating a rich set of text representations from the literature (Agrawal et al. 2018; Araque et al. 2017; Bravo-Marquez et al. 2016; Devlin et al. 2019; Felbo et al. 2017; Mikolov et al. 2018, 2013; Nguyen et al. 2020; Pennington et al. 2014; Tang et al. 2014; Xu et al. 2018; Zhu et al. 2015). To achieve a good overview of the static representations, we conduct an experimental evaluation in the sentiment analysis task with five different classifiers and 22 datasets.

RQ2 Considering state-of-the-art Transformer-based autoencoder models, which are the most effective in the sentiment classification of tweets? Regarding recent advances in language modeling, Transformer-based architectures have achieved state-of-the-art performances in many NLP tasks. Specifically, BERT (Devlin et al. 2019) is the first method that successfully uses the encoders components of the Transformer architecture (Vaswani et al. 2017) to learn contextualized embeddings from texts. Shortly after that, RoBERTa (Liu et al. 2019) was introduced by Facebook as an extension of BERT that uses an optimized training methodology. Next, BERTweet (Nguyen et al. 2020) was proposed as an alternative to RoBERTa for NLP tasks focusing on tweets. While RoBERTa was trained on traditional English texts, such as Wikipedia, BERTweet was trained from scratch using a massive corpus of 850M English tweets. In this context, to answer this research question, we conduct an experimental evaluation of BERT, RoBERTa, and BERTweet models in the sentiment analysis task with five different classifiers and 22 datasets to obtain a comprehensive analysis of their predictive performances. By evaluating these models we may obtain a robust overview of the Transformer-based autoencoder representations that better fit tweet's style.

RQ3 Can a second phase of continuous pretraining the Transformer-based autoencoder models using a large set of English tweets improve the sentiment classification performance? One of the benefits of pre-trained language models, such as the

Transformer-based models exploited in this study, is the possibility to adjust the language model to a specific domain. We aim at assessing whether the sentiment analysis of tweets can benefit from adapting BERT, RoBERTa and BERTweet language models to a vast, generic, and unlabeled set of around 6.7M English tweets from distinct domains. To that, we employed a second phase of training the pre-trained language model using the intermediate masked-language model task. Besides, considering that the adaptation procedure can be a very data-intensive task that may demand a lot of computational power, in addition to the large corpus of 6.7M tweets, we use in that process nine other samples with different sizes, varying from 500 to 1.5M tweets. We conduct an experimental evaluation with all models in the sentiment analysis task with five different classifiers and 22 datasets as in the previous questions.

RQ4 Can Transformer-based autoencoder models benefit from a second phase of adaptive pretraining procedure with tweets specific from sentiment analysis datasets? Although using unlabeled generic tweets to adjust a language model seems to be promising regarding the availability of data, we believe that the downstream sentiment task may benefit from the sentiment information that tweets from labeled datasets contain. In this context, we aim at identifying whether adjusting the language models with positive and negative tweets can boost the sentiment classification of tweets. We perform this evaluation by assessing three distinct strategies in order to simulate three real-world situations, as follows. In the first strategy, we use a specific sentiment dataset itself as the target domain dataset to adapt the language model. The second strategy simulates the case where a collection of general sentiment dataset is available to adapt the language model. In the third and last strategy, we combine the two previous situations. In short, we put together tweets from a target dataset and from a collection of sentiment datasets in the adaptation procedure. Finally, we present a comparison between the predictive performances achieved by these three evaluations and the adapted models evaluated in RQ3. As in the previous questions, we conduct the experiments with five different classifiers and 22 datasets.

In summation, given the large number of language models exploited in this study, our main contributions are: (i) a comparative study of a rich collection of publicly available static representations generated from distinct deep learning methods, and with different dimensions, vocabulary size, and from various kinds of corpora; (ii) an assessment of state-of-the-art contextualized language models from the literature, that is, Transformer-based autoencoder models, including BERT, RoBERTa, and BERTweet; (iii) an evaluation of distinct strategies for adapting Transformer-based autoencoder language models; and (iv) a general comparison over static, Transformer-based autoencoder, and adapted language models, aiming at determining the most suitable ones for detecting the sentiment expressed in tweets.[3]

In order to present our contributions, we organized this article as follows. Section 2 presents a literature review related to the language models examined in this study. In Sect. 3, we describe the experimental methodology we followed in the computational experiments, which are reported in Sects. 4, 5, 6, and 7, responding the four research

[3] The code and the detailed computational results from our investigation are publicly available at https://github.com/MeLL-UFF/tuning_sentiment.

question, respectively. Finally, in Sect. 8, we present the conclusions and directions for future research.

2 Literature review

Sentiment analysis is an automated process used to predict people's opinions, sentiments, evaluations, appraisals, attitudes, and emotions towards entities such as products, services, organizations, individuals, issues, events, topics, and their attributes (Liu 2020). Recently, sentiment analysis has been recognized as a suitcase research problem (Cambria et al. 2017), which involves solving different NLP classification sub-tasks, including sarcasm, subjectivity, and polarity detection, which is the focus of this study.

Pioneer works in the sentiment classification of tweets mainly focused on the polarity detection task, which aims at categorizing a piece of text as carrying a positive or negative connotation. For example, Go et al. (2009) define sentiment as a personal positive or negative feeling. There, they used unigrams as features to train different machine learning classifiers, using tweets with emoticons as training data. The unigram model, or Bag-of-Words (BoW), is the most basic representation in text classification problems.

Over the years, different techniques have been developed in NLP literature in an effort to make natural language easily processable by computers. Vector Space Models (VSMs) (Salton et al. 1975) are one of the earliest strategies used to represent the knowledge extracted from a given corpus. Earlier approaches to build VSMs are grounded on count-based methods, such as BoW with TF-IDF representation, which measures how important a word is to a document, relying on its frequency of occurrence in a corpus (Manning et al. 2008).

The BoW model, which assumes word order is not important, is based on the hypothesis that the frequencies of words in a document tend to indicate the relevance of the document to a query (Salton et al. 1975). This hypothesis expresses the belief that a column vector in a term-document matrix captures an aspect of the meaning of the corresponding document or phrase. Precisely, Let X be a term-document matrix. Suppose the document collection contains n documents and m unique terms. The matrix X will then have m rows (one row for each unique term in the vocabulary) and n columns (one column for each document). Let w_i be the i-th term in the vocabulary and let d_j be the j-th document in the collection. The i-th row in X is the row vector $\mathrm{x}_{i:}$ and the j-th column in X is the column vector $\mathrm{x}_{:j}$. The row vector $\mathrm{x}_{i:}$ contains n elements, one element for each document, and the column vector $\mathrm{x}_{:j}$ contains m elements, one element for each term. Suppose X is a simple matrix of frequencies, then the element x_{ij} in X is the frequency of the i-th term w_i in the j-th document d_j (Turney and Pantel 2010).

Such a simple way of creating numeric representations from texts have motivated early studies in detecting the sentiment expressed in tweets (Barbosa and Feng 2010; Go et al. 2009; Pak and Paroubek 2010). However, though widely adopted, this kind of feature representation leads to the curse of dimensionality due to the large number of uncommon words tweets contain (Saif 2015).

Thus, with the revival and success of neural-based learning techniques, several methods that learn dense real-valued low dimensional vectors to represent words have been proposed, such as Word2Vec (Mikolov et al. 2013), FastText (Mikolov et al. 2018), and GloVe (Pennington et al. 2014). Word2Vec (Mikolov et al. 2013) is one of the pioneer models to become popular taking advantage from the development of neural networks over the years. Wor2Vec is actually a software package composed of two distinct implementations of language-models, both based on a feed-forward neural architecture, namely Continuous Bag-Of-Words (CBOW) and Skip-gram. The CBOW model aims at predicting a word given its surrounding context words. Conversely, the Skip-gram model predicts the words in the surrounding context given a target word. Both architectures consist of input, a hidden layer and an output layer. The input layer has the size of the vocabulary and encodes the context by combining the one-hot vector representations of surrounding words of a given target word. The output layer has the same size as the input layer and contains a one-hot vector of the target word obtained during the training. However, one of the main disadvantages of those models is that they usually struggle to deal with out-of-vocabulary (OOV) words, i.e., words that have not been seen in the training data before. To address this weakness, more complex approaches have been proposed, such as FastText (Mikolov et al. 2018).

FastText (Mikolov et al. 2018) is based on the Skip-gram model (Mikolov et al. 2013), still it considers each word as a bag of character n-grams, which are contiguous sequences of n characters from a word, including the word itself. A dense vector is learned to each character n-gram and the dense vector associated to a word is taken from the sum of those representations. Thus, FastText can deal with different morphological structure of words that covers the words not seen in the training phase, i.e., OOV words. For that reason, FastText is also able to deal with tweets, considering the huge number of uncommon and unique words in this kind of text.

Going to another direction, the GloVe model (Pennington et al. 2014) attempts at making efficient use of statistics of word occurrences in a corpus to learn better word representations. Pennington et al. (2014) present a model that rely on the insight that ratios of co-occurrences, rather than raw counts, encode semantic information about pair of words. This relationship is used to derive a suitable loss function for a log-linear model, which is then trained to maximize the similarity of every word pair, as measured by the ratios of co-occurrences. Given a probe word, the ratio can be small, large or equal to one depending on their correlations. This ratio gives hints on the relations between three different words. For example, given a probe word and two others w_i and w_j, if the ratio is large, the probe word is related to w_i but not w_j.

In general, methods for learning word embeddings deal well with the syntactic role of words but ignore the potential sentiment they carry. In the context of sentiment analysis, words with similar syntactic role but opposite sentiment polarity, such as *good* and *bad*, are usually mismapped to neighbouring word vectors. To address this issue, Tang et al. (2014) proposed the Sentiment-Specific Word Embedding model (SSWE), which encodes the sentiment information in the embeddings. Specifically, they developed neural networks that incorporate the supervision from sentiment polarity of texts in their loss function. To that, they slide the window of n-gram across a sentence, and then predict the sentiment polarity based on each n-gram with a shared neural network. In addition to SSWE, other methods have been proposed in order to

improve the quality of word representations in sentiment analysis, by leveraging the sentiment information in the training phase, such as DeepMoji (Felbo et al. 2017), Emo2Vec (Xu et al. 2018), and EWE (Agrawal et al. 2018).

The aforementioned word embedding models have been used as standard components in most sentiment analysis methods. However, they pre-compute the representation for each word independently from the context they are going to appear. This static nature of these models results in two problems: (i) they ignore the diversity of meaning each word may have, and (ii) they suffer from learning long-term dependencies of meaning. Different from those static word embedding techniques, *contextualized embeddings* are not fixed, adapting the word representation to the context it appears. Precisely, at training time, for each word in a given input text, the learning model analyzes the context, usually using sequence-based models, such as recurrent neural networks (RNNs), and adjusts the representation of the target word by looking at the context. These context-awareness embeddings are actually the internal states of a deep neural network trained in an self-supervised setting. Thus, the training phase is carried out independently from the primary task on an extensive unlabeled data. Depending on the sequence-based model adopted, these contextualized models can be divided into two main groups, namely RNN-based (Peters et al. 2018) and Transformers-based (Lan et al. 2020; Liu et al. 2019; Nguyen et al. 2020).

Transfer learning strategies have also emerged to improve the quality of word representation, such as ULMFit (Universal Language Model Fine-tuning) (Howard and Ruder 2018). ULMFit is an effective transfer learning method that can be applied to any NLP task, and introduces key techniques for fine-tuning a language model, consisting of three stages, described as follows. First, the language model is trained on a general-domain corpus to capture generic features of the language in different layers. Next, the full language model is fine-tuned on the target task data using discriminative fine-tuning and slanted triangular learning rates (STLR) to learn task-specific features. Lastly, the model is fine-tuned on the target task using gradual unfreezing and STLR to preserve low-level representations and to adapt high-level ones.

Fine-tuning techniques made possible the development and availability of pre-trained contextualized language models using massive amounts of data. For example, Peters et al. (2018) introduced ELMo (Embeddings from Language Models), a deep contextualized model for word representation. ELMo comprises a Bi-directional Long-Short-Term-Memory Recurrent Neural Network (BiLSTM) to combine a forward model, looking at the sequence in the traditional order, and a backward model, looking at the sequence in the reverse order. ELMo is composed of two layers of BiLSTM sequence encoder responsible for capturing the semantics of the context. Besides, some weights are shared between the two directions of the language modeling unit and there is also a residual connection between the LSTM layers to accommodate the deep connections without the gradient vanishing issue. ELMo also makes use of the character-based technique for computing embeddings. Therefore, it benefits from the characteristics of character-based representations to avoid OOV words.

Although ELMo is more effective as compared to static pre-trained models, its performance may be degraded when dealing with long texts, exposing a trade-off between efficient learning by gradient descent and latching on information for long periods (Bengio et al. 1994). Transformers-based language models, on the other hand,

have been proposed to solve the gradient propagation problems described in (Bengio et al. 1994). Compared to RNNs, which process the input sequentially, Transformers work in parallel, which brings benefits when dealing with large corpora. Moreover, while RNNs by default process the input in one direction, Transformers-based models can attend to the context of a word from distant parts of a sentence and pay attention to the part of the text that really matters, using self-attention (Vaswani et al. 2017).

The OpenAI Generative Pre-Training Transformer model (GPT) (Radford et al. 2018) is one of the first attempts to learn representations using Transformers. It encompasses only the decoder component of the Transformer architecture with some adjustments, discarding the encoder part. Therefore, instead of having a source and a target sentence for the sequence transduction model, a single sentence is given to the decoder. GPT' objective function targets at predicting the next word given a sequence of words, as a standard language modeling goal. To comply with the standard language model task, while reading a token, GPT can only attend to previously seen tokens in the self-attention layers. This setting can be limiting for encoding sentences, since understanding a word might require processing the ones coming after it in the sentence.

Devlin et al. (2019) addressed the unidirectional nature of GPTs by presenting an strategy called BERT (Bidirectional Encoder Representations from Transformers) that, as the name says, encodes sentences by looking them at both directions. BERT is also based on the Transformer architecture but, contrary to the GPT, it is based on the encoder component of that architecture. The essential improvement over GPT is that BERT provides a solution for making Transformers bidirectional by applying masked language models, which randomly masks some percentage of the input tokens, and the objective is to predict those masked tokens based on their context. Also, in (Devlin et al. 2019), they use a next sentence prediction task for predicting whether two text segments follow each other. All those improvements have made BERT to achieve state-of-the-art results in various NLP tasks when it was published.

Later, Liu et al. (2019) proposed RoBERTa (Robustly optimized BERT approach), achieving even better results than BERT. RoBERTa is an extension of BERT with some modifications, such as: (i) training the model for a longer period of time, with bigger batches, over more data, (ii) removing the next sentence prediction objective, (iii) training on longer sequences, and (iv) dynamically changing the masking pattern applied to the training data.

Recently, Nguyen et al. (2020) introduced BERTweet, an extension of RoBERTa trained from scratch with tweets. BERTweet has also the same architecture as BERT, but it is trained using the same Roberta pre-training procedure instead. BERTweet consumes a corpus of 850M English tweets, which is a concatenation of two corpora. The first corpus contains 845M English tweets from the Twitter Stream dataset and the second one contains 5M English tweets related to the COVID-19 pandemic. In (Nguyen et al. 2020), the proposed BERTweet model outperformed RoBERTa baselines in some tasks on tweets, including sentiment analysis.

As far as we know, most studies in language modeling focus on designing new effective models in order to improve the predictive performance of distinct NLP tasks. For example, Devlin et al. (2019) and Liu et al. (2019) have respectively introduced BERT and RoBERTa, which achieved state-of-the-art results in many NLP tasks.

Nevertheless, they did not evaluate the performance of such methods on the sentiment classification of tweets. Nguyen et al. (2020), on the other hand, used only a unique generic collection of tweets when evaluating their BERTweet strategy. In this context, we fulfill a robust evaluation of existing language models from distinct natures, including static representations, Transformer-based autoencoder models, and fine-tuned models, by using a significant set of 22 datasets of tweets from different domains and sizes. In the following sections, we present the assessment of such models.

3 Experimental methodology

This section presents the experimental methodology we followed in this article. We begin by describing, in Sect. 3.1, the twenty-two benchmark datasets used to evaluate the different language models we investigate in this study. In Sect. 3.2, we present the experimental protocol we followed. Then, in Sect. 3.3, we describe the computational experiments reported in Sects. 4, 5, 6, and 7.

3.1 Datasets

We used a large set of twenty-two datasets[4] (Carvalho and Plastino 2021) to assess the effectiveness of the distinct word representation models described in Sect. 2. Table 1 summarizes the main characteristics of these datasets, namely the abbreviation we use when reporting the experimental results to save space (*Abbrev.* column), the domain they belong (*Domain* column), number of positive tweets (#*pos.* column), proportion of positive tweets (%*pos.* column), number of negative tweets (#*neg.* column), proportion of negative tweets (%*neg.* column), and the total number of tweets (*Total* column).

Those datasets have been extensively used in the literature of Twitter sentiment analysis and we believe they provide a diverse scenario in evaluating embeddings of tweets in the sentiment classification task, regarding a variety of domains, sizes, and class balance. For example, while datasets SemEval13, SemEval16, SemEval17, and SemEval18 contain generic tweets, other datasets, such as iphone6, movie, and archeage, contain tweets of a particular domain. Also, the datasets vary a lot in size, with some of them containing only dozens of tweets, such as irony and sarcasm. We believe that this diverse and large collection of datasets may help drawing more concise and robust conclusions on the effectiveness of distinct language models in the sentiment analysis task.

3.2 Experimental protocol

To assess the effect of different kinds of word representation models in the polarity classification task, we follow the protocol of first extracting the features from the

[4] The datasets are publicly available at https://github.com/joncarv/air-datasets.

Table 1 Characteristics of the Twitter sentiment datasets ordered by size (Total column)

Dataset	Abbrev.	Domain	#pos.	%pos.	#neg.	%neg.	Total
irony (Gonçalves et al. 2015)	iro	Irony	22	34%	43	66%	65
sarcasm (Gonçalves et al. 2015)	sar	Sarcasm	33	46%	38	54%	71
aisopos[5]	ntu	Generic	159	57%	119	43%	278
SemEval-Fig[6]	S15	Irony/Metaphors	47	15%	274	85%	321
sentiment140 (Go et al. 2009)	stm	Generic	182	51%	177	49%	359
person (Chen et al. 2012)	per	Towards a Person	312	71%	127	29%	439
hobbit (Lochter et al. 2016)	hob	Movies	354	68%	168	32%	522
iphone6 (Lochter et al. 2016)	iph	Products	371	70%	161	30%	532
movie (Chen et al. 2012)	mov	Movies	460	82%	101	18%	561
sanders[7]	san	Business	570	47%	654	53%	1,224
Narr (Narr et al. 2012)	Nar	Generic	739	60%	488	40%	1,227
archeage (Lochter et al. 2016)	arc	Games	724	42%	994	58%	1,718
SemEval18 (Mohammad et al. 2018)	S18	Equity Evaluation Corpus	865	47%	994	53%	1,859
OMD (Diakopoulos and Shamma 2010)	OMD	Presidential Debate	710	37%	1,196	63%	1,906
HCR (Speriosu et al. 2011)	HCR	Health Care Reform	539	28%	1,369	72%	1,908
STS-gold (Saif et al. 2013)	STS	Generic	632	31%	1,402	69%	2,034
SentiStrength (Thelwall et al. 2012)	SSt	Generic	1,340	59%	949	41%	2,289
Target-dependent (Dong et al. 2014)	Tar	Celebrities	1,734	50%	1,733	50%	3,467
Vader (Hutto and Gilbert 2014)	vad	Generic	2,897	69%	1,299	31%	4,196
SemEval13[8]	S13	Generic	3,183	73%	1,195	23%	4,378
SemEval17 (Rosenthal et al. 2017)	S17	Generic	2,375	37%	3,972	63%	6,347
SemEval16 (Nakov et al. 2016)	S16	Generic	8,893	73%	3,323	27%	12,216

several vector-based language[5,6,7,8] representation mechanisms (BOW, static embeddings, contextualized embeddings). Next, those features compose the input attribute space for five distinct classifiers, namely Support Vector Machine (SVM), Logistic Regression (LR), Random Forest (RF), XGBoost (XBG), and Multi-layer Perceptron (MLP). We adopted scikit-learn's[9] implementations of those machine learning algorithms. Although we have used the default parameters in most of the cases, it is

[5] http://www.grid.ece.ntua.gr.

[6] http://www.alt.qcri.org/semeval2015/task11.

[7] https://www.github.com/karanluthra/twitter-sentiment-training.

[8] https://www.cs.york.ac.uk/semeval-2013/task2.html.

[9] https://scikit-learn.org.

important to mention that we set the class balance parameter for SVM, LR, and RF (*class_weight* = *balanced*). Also, for LR, we set the maximum number of iterations to 500 (*max_iter* = 500) and the solver parameter to *liblinear*. Moreover, for MLP, we set the number of hidden layers to 100. We aim at determining which word representation models are the most effective ones in Twitter sentiment analysis by leveraging different types of classifiers, thus examining how they deal with the peculiarities of each evaluated model. Furthermore, it is important to note that we do not aim at establishing the best classifier for the sentiment analysis task, which may require a specific study and additional computational experiments.

Preprocessing is the first step in many text classification problems and the use of appropriate techniques can reduce noise hence improving classification effectiveness (Fayyad et al. 2003). As this manuscript's main goal is to evaluate the performance of different models of tweet representation, the preprocessing step is simple so that the focus is on the word representation models and classifiers. Thus, for each tweet in a given dataset, we only replace URLs by the token *someurl*, user mentions by the token *someuser*, and all tokens were lowercased.

In the experimental evaluation, the predictive performance of the sentiment classification is measured in terms of accuracy and F_1-macro. Precisely, for each evaluated dataset, the accuracy of the classification was computed as the ratio between the number of correctly classified tweets and the total number of tweets, following a stratified ten-fold cross-validation. F_1-macro was computed as the unweighted average of the F_1-score for the positive and negative classes. Moreover, all experiments were performed by using a Tesla P100-SXM2 GPU within Ubuntu operating system, running in a machine with Intel(R) Xeon(R) CPU E5-2698 v4 processor.

Lastly, as recommended by Demšar (2006), we ran the Friedman test followed by the Nemenyi post-hoc test to determine whether the differences among the results are statistically significant at a 0.05 significance level. Whenever applicable, we present the results of the statistical tests immediately below each results table. We use the symbol $\succ$ to show that a word representation model x is significantly better than another word representation model y, so that $\{x\} \succ \{y\}$.

3.3 Computational experiments details

In the next sections, we evaluate a significant collection of vector-based word representation models attempting to answer the research questions introduced in Sect. 1. Specifically, we conduct a comparative study of vector-based word representation models from distinct natures, including Bag of Words, as a classic baseline, static representations and representations induced from Transformer-based autoencoder models, by including a second-phase training or not the intermediate masked language task, in order to acknowledge their effectiveness in the polarity classification of English tweets. These language representation models are incrementally evaluated throughout Sects. 4, 5, 6, and 7.

In Sect. 4, we begin by analyzing the predictive performance of the static representations, which include 13 pretrained embeddings from the literature, as shown in Table 2, as well as the classical BOW with TF-IDF representation schema. Regarding

Table 2 Characteristics of the static pretrained embeddings ordered by the number of dimensions

Embedding	$\|D\|$	$\|V\|$	Architecture	Corpus
SSWE (Tang et al. 2014)	50	137K	Feed-forward network	Twitter (10M tweets)
Emo2Vec (Xu et al. 2018)	100	1.2M	Convolutional network	Twitter (1.9M tweets)
GloVe-TWT (Pennington et al. 2014)	200	1.2M	log-bilinear model	Twitter (27B tokens)
DeepMoji (Felbo et al. 2017)	256	50K	Recurrent network	Twitter (1B tweets)
EWE (Agrawal et al. 2018)	300	183K	Recurrent network	Amazon reviews (200K reviews)
GloVe-WP (Pennington et al. 2014)	300	400K	log-bilinear model	Wikipedia/Gigaword (6B tokens)
FastText (Mikolov et al. 2018)	300	1M	Feed-forward network	Wikipedia/web pages/news (16B tokens)
w2v-GN (Mikolov et al. 2013)	300	3M	Feed-forward network	Google news (100B tokens)
w2v-Edin (Bravo-Marquez et al. 2016)	400	259K	Feed-forward network	Twitter (10M tweets)
w2v-Araque (Araque et al. 2017)	500	57K	Feed-forward network	Twitter (1.28M tweets)
BERT (Devlin et al. 2019)	768	30K	Transformers	BooksCorpus (Zhu et al. 2015)/Eng. Wiki (3.3B words)
RoBERTa (RoB) (Zhu et al. 2015)	768	50K	Transformers	5 datasets (Liu et al. 2019) (161GB)
BERTweet (BTWT) (Nguyen et al. 2020)	768	64K	Transformers	Twitter (850M tweets)

the static embeddings described in Table 2, we have selected representations trained on distinct kinds of texts (Corpus column) and built from different architectures (Architecture column), from feedforward neural networks to Transformer-based ones. The $|D|$ and $|V|$ columns refer to the dimension and vocabulary size of each pretrained embedding, respectively. Although the most usual way of employing embeddings trained from Transformer-based architectures is running the text through the model to obtain contextualized representations, here we first investigate how these models behave when the experimental protocol is the same as earlier embeddings models: pretrained embeddings are collected from the embeddings layer and are the input of the classifiers.

Next, in Sect. 5, we present an evaluation of state-of-the-art Transformer-based autoencoder models, including BERT (Devlin et al. 2019), RoBERTa (Liu et al. 2019), and BERTweet (Nguyen et al. 2020). To achieve a proper vector representation for each sentence, first we get the last four layers of the model for each token of the sentence and concatenate them, generating a 3072-dimension (4×768) representation for each token. Then, to build the sentence embedding, we take the average of these token vector representations. For the sake of simplicity, the Transformer-based autoencoder models assessed in this study are referred to hereafter as Transformer-based models.

Lastly, in Sects. 6 and 7, we evaluate the effectiveness of adapting the aforementioned Transformer-based models regarding the intermediate masked-language task in two different ways: (i) by using a huge collection of unlabeled, or non-sentiment, tweets, and (ii) by using tweets from sentiment datasets.

In Sect. 6, regarding the non-sentiment adaptation approach, we adopted the general purpose collection of unlabeled tweets from the Edinburgh corpus (Petrović et al. 2010), which contains 97M tweets in multiple languages. Tweets written in languages other than English were discarded, resulting in a final corpus of 6.7M English tweets, which was then used to adapt BERT, RoBERTa, and BERTweet. In addition to the entire corpus of 6.7M tweets, we used nine other samples with different sizes, varying from 500 to 1.5M tweets. Specifically, we generated samples containing 500 (0.5K), 1K, 5K, 10K, 25K, 50K, 250K, 500K, and 1.5M non-sentiment tweets.

Conversely, in Sect. 7, we evaluated the sentiment adaptation procedure using positive and negative tweets from the twenty-two benchmark datasets described in Table 1. For this purpose, we used each dataset once as the target dataset, while the others were used as the source datasets. More clearly, for each assessed dataset, referred to as the target dataset, we explored three distinct strategies to adapt the masked-language model: (i) by using only the tweets from the target sentiment dataset itself, (ii) by using the tweets from the remaining 21 datasets, and (iii) by using the entire collection of tweets from the 22 datasets, including the tweets from the target dataset.

4 Evaluation of static text representations

The computational experiments conducted in this section aim at answering the research question RQ1, as follows:

RQ1. Which static embeddings are the most effective in the sentiment classification of tweets?

We answer this question by assessing the predictive power of the 13 pretrained embeddings described in Table 2. These embeddings were generated from distinct neural networks architectures, with different dimensions and vocabulary size, and trained on various kinds of corpora. Recall that by static embeddings we mean that the features are gathered from the embeddings layer working as a look-up table of tokens. In addition to the pretrained embeddings, we evaluate the BoW model with the TF-IDF representation, which is the most basic text representation used in Twitter sentiment analysis and text classification tasks in general. For all tweet representation, we take the average of all tokens representation of the tweet.

We begin by evaluating the predictive performance of the static representations for each classification algorithm. To limit the number of tables in the manuscript, we report the computational results in detail for SVM as an example of this evaluation.[10] Tables 3 and 4 show the results achieved by using each static representation to train an SVM classifier, in terms of classification accuracy and unweighted F_1-macro, respectively. The boldfaced values indicate the best results, and the last three lines show the total

[10] The detailed computational results for each classifier are publicly available at https://github.com/MeLL-UFF/tuning_sentiment.

number of wins for each static representation (#wins row), as well as a ranking of the results (rank sums and position rows). Precisely, for each dataset, we assign scores, from 1.0 to 14.0, to each assessed representation (each column), in ascending order of accuracy (F_1-macro), where the score 1.0 is assigned to the representation with the highest accuracy (F_1-macro). Thus, low score values indicate better results. When two assessed representation has the same performance, we take an average of their scores. If two assessed representations achieve the best performance, they will receive a score of 1.5 ((1+2)/2). Finally, we sum up the assigned scores obtained in each dataset for each assessed representation to calculate rank sums. With the rank sum of each assessed representation, we rank the rank-sum result from the best (1) to the worst (14), calculating the rank position.

As we can see in Tables 3 and 4, RoBERTa (RoBstatic column) achieved the best performance in nine out of the 22 datasets in terms of accuracy, in 11 out of the 22 datasets in terms of F_1-macro, and was ranked first in the overall evaluation (position row). Regarding the number of wins (#wins row), we can note that Emo2Vec and SSWE achieved the second best results, reaching the best performance in four out of the 22 datasets for both accuracy and F_1-macro. However, regarding the overall evaluation (position row), w2v-Edin and w2v-GN were ranked among the top three best static representations along with RoBERTa, in terms of accuracy. Regarding F_1-macro, the top three best static representations were RoBERTa, w2v-Edin and BERT (BERT-static column). Finally, the Friedman test followed by the Nemenyi post-hoc test detected that the top three best representations – RoBERTa, w2v-Edin, w2v-GN in terms of accuracy, and RoBERTa, w2v-Edin, BERT in terms of F_1-macro – are significantly better than many of the other static representations, as shown below Tables 3 and 4. Nevertheless, there is no significant difference between them.

Tables 5 and 6 show a summary of the results by evaluating each static representation on the 22 datasets, for each classification algorithm. Each cell indicates the number of wins, the rank sums, and the rank position achieved by the related static representation (each line) used to train the corresponding classifier (each column). The *Total* column indicates the total number of wins, the total rank sums, and the total rank position, i.e., the sum of the rank positions presented in each cell for each assessed model. Moreover, in the total column, we underline the top three best overall results in terms of total rank position.

Regarding the overall evaluation (Total column), from Tables 5 and 6, we can see that although Emo2Vec achieved the highest total number of wins (i.e., 27 wins in terms of accuracy, and 29 wins in terms of F_1-macro), w2v-Edin was ranked as the best overall model, achieving the lowest total rank position for both accuracy (22.0) and F_1-macro (21.0). Nevertheless, considering each classifier (each column), we can note that RoBERTa achieved the best performance when used to train LR, SVM, and MLP, for both accuracy and F_1-macro. Conversely, Emo2Vec achieved the best overall results when used to train RF and XGB classifiers. Analyzing the overall results in terms of the total rank position (Total column), we observe that Emo2Vec and w2v-GN, along with w2v-Edin, are ranked as the top three best static representations. These results suggest that w2v-Edin, Emo2Vec, and w2v-GN are well-suited static representations for Twitter sentiment analysis.

Table 3 Accuracies (%) achieved by evaluating the static representations using the SVM classifier

Dataset	w2v-GN	GloVe-WP	FastText	EWE	GloVe-TWT	BERT static	TF-IDF	w2v-Araque	w2v-Edin	SSWE	Emo2Vec	DeepMoji	RoB static	BTWT static
iro	70.95	69.52	67.86	52.14	51.67	60.24	64.52	63.10	66.43	70.48	**78.33**	61.90	70.48	39.29
sar	68.75	68.75	61.61	68.75	67.32	70.54	54.64	71.79	68.93	**87.32**	73.04	67.50	56.07	57.68
ntu	79.15	88.47	73.04	70.54	73.70	83.47	74.40	77.71	93.15	**93.53**	81.28	91.73	89.91	81.98
S15	87.88	83.20	81.95	88.80	83.50	89.72	87.23	87.54	89.41	77.56	82.88	88.16	**90.34**	87.52
stm	85.22	83.02	83.56	83.01	83.56	81.06	81.59	82.71	**87.74**	80.75	85.24	81.34	84.94	71.59
per	**83.37**	76.75	76.77	74.69	73.56	73.32	80.18	74.93	79.03	72.44	76.09	76.98	80.40	69.23
hob	89.66	85.24	83.51	90.22	79.88	91.75	**93.29**	91.37	90.04	76.99	87.18	91.57	92.90	90.03
iph	78.00	75.57	76.13	72.17	73.88	79.33	80.07	80.46	78.01	76.49	**81.57**	78.56	80.08	76.51
mov	82.90	76.66	76.65	75.95	71.67	83.96	84.67	83.78	84.14	78.09	83.07	81.29	**85.38**	79.35
san	82.59	78.50	80.55	80.54	80.38	81.37	83.33	81.46	83.25	80.14	81.12	80.80	**83.57**	78.84
Nar	84.51	83.94	84.03	83.78	85.33	83.46	80.36	84.18	88.67	**88.84**	88.34	88.83	87.21	80.61
arc	85.45	83.59	85.16	84.05	85.27	86.61	**87.54**	85.16	86.67	79.63	83.35	86.09	87.43	84.87
S18	80.37	78.48	80.69	80.26	80.20	84.56	82.09	75.79	82.36	81.28	80.96	78.97	**86.50**	79.51
OMD	83.79	82.21	81.42	77.55	77.38	84.15	79.22	77.70	82.84	76.50	75.76	77.02	**85.10**	82.37
HCR	74.78	72.58	72.64	71.59	73.21	76.36	**80.24**	73.95	75.94	67.29	70.54	73.21	78.30	73.27
STS	85.94	83.19	85.74	84.71	84.90	86.97	83.97	86.92	87.02	**88.99**	86.19	88.69	87.86	83.09
SSt	77.98	75.10	76.98	77.76	77.24	78.51	73.48	76.28	79.56	79.77	**84.93**	79.64	80.21	73.00

Table 3 continued

Dataset	w2v-GN	GloVe-WP	FastText	EWE	GloVe-TWT	BERT static	TF-IDF	w2v-Araque	w2v-Edin	SSWE	Emo2Vec	DeepMoji	RoB static	BTWT static
Tar	83.67	81.92	83.82	83.18	82.81	83.44	82.35	81.40	83.39	79.18	82.98	82.90	**84.42**	80.36
Vad	88.25	84.51	87.01	86.44	85.96	87.51	83.27	85.30	87.73	85.70	85.94	86.77	**89.32**	81.82
S13	81.52	78.53	79.49	78.39	78.69	81.59	80.52	79.19	80.31	81.04	**87.80**	81.89	83.60	77.64
S17	88.61	86.34	88.29	87.22	87.71	88.47	87.95	84.53	87.96	81.63	85.60	85.47	**89.03**	86.01
S16	84.50	82.51	84.50	83.15	83.87	85.27	85.85	81.37	84.39	78.51	82.29	82.79	**86.00**	81.69
#wins	1	0	0	0	0	0	3	0	1	4	4	0	**9**	0
rank sums	116.5	221.0	183.5	212.0	216.0	117.0	155.0	185.5	95.0	200.5	154.0	152.5	**56.5**	245.0
position	3.0	13.0	8.0	11.0	12.0	4.0	7.0	9.0	2.0	10.0	6.0	5.0	**1.0**	14.0

{RoB static} $\succ$ {BTWT static, DeepMoji, EWE, Emo2Vec, FastText, Glove-TWT, Glove-WP, SSWE, TF-IDF, w2v-Araque}
{w2v-Edin} $\succ$ {BTWT static, EWE, GloVe-TWT, GloVe-WP, SSWE}
{w2v-GN} $\succ$ {BTWT static, EWE, GloVe-TWT, GloVe-WP}
Bold values indicate the best results

Table 4 F_1-macro scores (%) achieved by evaluating the static representations using the SVM classifier

Dataset	w2v-GN	GloVe-WP	FastText	EWE	GloVe-TWT	BERT static	TF-IDF	w2v-Araque	w2v-Edin	SSWE	Emo2Vec	DeepMoji	RoB static	BTWT static
iro	60.03	64.46	60.78	47.51	49.40	54.84	39.11	48.35	58.24	67.04	**72.17**	53.46	61.16	35.48
sar	66.00	67.75	60.22	67.55	63.32	69.29	51.68	69.97	66.70	**86.93**	70.72	64.72	50.28	52.69
ntu	78.79	87.81	72.46	70.19	73.02	83.15	71.55	76.72	92.99	**93.29**	80.99	91.53	89.42	81.22
S15	69.91	65.74	68.13	75.02	70.93	77.04	58.51	66.33	77.62	67.16	70.24	73.65	**78.48**	75.38
stm	85.18	82.94	83.46	82.96	83.51	80.96	81.46	82.60	**87.67**	80.71	85.18	81.28	84.87	71.50
per	**80.58**	74.31	74.86	73.03	71.80	70.85	72.27	71.59	76.97	70.46	74.16	74.46	77.30	67.53
hob	88.56	83.89	82.25	89.26	78.71	90.92	91.73	90.52	88.95	75.49	86.03	90.73	**92.08**	89.05
iph	75.96	74.31	74.96	71.00	72.63	77.34	71.55	77.92	76.18	74.93	**79.97**	76.62	78.00	74.70
mov	73.42	68.28	68.39	67.23	64.17	73.72	58.15	73.55	**76.15**	71.60	76.06	72.97	73.86	68.37
san	82.38	78.24	80.28	80.32	80.03	81.20	82.99	81.33	83.10	79.99	80.98	80.60	**83.41**	78.43
Nar	84.08	83.50	83.76	83.42	85.02	82.93	79.32	83.72	88.38	**88.48**	88.01	88.45	86.82	80.06
arc	85.14	83.27	84.84	83.80	84.87	86.37	87.11	84.90	86.34	78.88	82.84	85.61	**87.19**	84.56
S18	80.21	78.31	80.51	80.03	79.91	84.44	81.55	75.44	82.15	81.13	80.86	78.74	**86.40**	79.24
OMD	82.50	80.91	80.03	76.00	75.81	82.70	76.57	76.01	81.36	74.86	74.28	75.71	**83.85**	80.63
HCR	70.97	68.96	69.78	68.26	70.44	72.94	72.16	69.17	72.38	62.99	66.96	68.79	**74.55**	69.74
STS	84.12	81.29	83.99	82.93	83.26	84.99	79.56	84.95	85.20	**87.62**	84.49	87.08	85.95	80.67
SSt	77.69	74.90	76.69	77.59	77.07	78.20	72.49	75.89	79.27	79.49	**84.68**	79.36	79.83	72.74

Table 4 continued

Dataset	w2v-GN	GloVe-WP	FastText	EWE	GloVe-TWT	BERT static	TF-IDF	w2v-Araque	w2v-Edin	SSWE	Emo2Vec	DeepMoji	RoB static	BTWT static
Tar	83.67	81.91	83.81	83.18	82.80	83.43	82.33	81.39	83.38	79.13	82.96	82.87	**84.42**	80.32
Vad	86.68	82.62	85.33	84.82	84.23	85.76	78.48	83.38	86.14	84.08	84.28	85.14	**87.80**	79.66
S13	78.43	75.83	76.82	75.52	76.10	78.67	72.24	75.95	77.64	78.40	**85.59**	79.16	80.40	74.63
S17	88.02	85.67	87.69	86.61	87.05	87.75	86.9	83.52	87.30	80.51	84.89	84.69	**88.37**	85.13
S16	81.58	79.52	81.62	80.15	81.10	82.48	81.68	78.17	81.63	75.82	79.54	80.00	**83.18**	78.62
#wins	1	0	0	0	0	0	0	0	2	4	4	0	**11**	0
rank sums	123.5	215.0	168.0	206.0	202.0	111.0	212.0	198.0	89.0	193.0	143.5	153.0	**56.0**	240.0
position	4.0	13.0	7.0	11.0	10.0	3.0	12.0	9.0	2.0	8.0	5.0	6.0	**1.0**	14.0

{RoB static} $\succ$ {BTWT static, DeepMoji, EWE, FastText, GloVe-TWT, GloVe-WP, SSWE, TF-IDF, w2v-Araque}
{w2v-Edin} $\succ$ {BTWT static, EWE, GloVe-TWT, GloVe-WP, SSWE, TF-IDF, w2v-Araque}
{BERT static} $\succ$ {BTWT static, EWE, TF-IDF, GloVe-WP}
Bold values indicate the best results

Table 5 Overview of the results (number of wins, rank sum, and rank position, respectively) achieved by evaluating each static representation on the 22 datasets, for each classification algorithm, in terms of accuracy

Representation	LR	SVM	MLP	RF	XGB	Total
w2v-GN	3/116.5/4.0	1/116.5/3.0	1/151.5/6.0	1/159.0/6.0	3/125.5/4.0	9/669.0/23.0
GloveWP	0/149.0/7.0	0/221.0/13.0	0/212.5/11.0	0/207.0/11.0	0/195.5/10.0	0/985.0/52.0
FastText	0/192.5/10.0	0/183.5/8.0	2/148.5/5.0	0/190.0/9.0	0/151.5/6.0	2/866.0/38.0
EWE	1/101.5/3.0	0/212.0/11.0	0/142.0/4.0	0/170.5/7.0	0/153.5/8.0	1/779.5/33.0
GloveTW	1/97.0/2.0	0/216.0/12.0	1/152.0/7.0	1/105.0/3.0	2/123.0/3.0	5/693.0/27.0
SSWE	4/152.0/8.0	4/200.5/10.0	1/234.0/14.0	6/79.5/2.0	4/153.0/7.0	19/819.0/41.0
TF-IDF	**5**/146.5/6.0	3/155.0/7.0	1/225.0/13.0	3/116.0/5.0	2/215.5/13.0	14/858.0/44.0
DeepMoji	0/174.0/9.0	0/152.5/5.0	1/167.5/8.0	0/171.0/8.0	1/144.0/5.0	2/809.0/35.0
w2v-Araque	0/204.0/11.0	0/185.5/9.0	0/221.0/12.0	0/202.5/10.0	0/214.5/12.0	0/1027.5/54.0
w2v-Edin	0/220.5/12.0	1/95.0/2.0	2/100.5/2.0	1/105.5/4.0	2/106.5/2.0	6/628.0/**22.0**
Emo2Vec	4/120.0/5.0	4/154.0/6.0	2/192.0/10.0	**10/46.0/1.0**	**7/98.5/1.0**	**27/610.5**/23.0
BERT-static	0/264.0/13.0	0/117.0/4.0	3/117.5/3.0	0/235.5/12.0	0/207.5/11.0	3/941.5/43.0
RoBERTa-static	**5/82.5/1.0**	**9/56.5/1.0**	**8/74.0/1.0**	0/244.0/13.0	1/167.5/9.0	23/624.5/25.0
BERTweet-static	0/290.0/14.0	0/245.0/14.0	0/172.0/9.0	0/278.5/14.0	0/254.0/14.0	0/1239.5/65.0

Bold values indicate the best results. Underline values indicate top three best overall results in terms of total rank position

Table 6 Overview of the results (number of wins, rank sum, and rank position, respectively) achieved by evaluating each static representation on the 22 datasets, for each classification algorithm, in terms of F_1-macro

Representation	LR	SVM	MLP	RF	XGB	Total
w2v-GN	3/113.0/4.0	1/123.5/4.0	1/146.5/5.0	1/161.5/6.0	3/129.0/4.0	9/673.5/23.0
GloveWP	0/152.0/6.5	0/215.0/13.0	1/213.0/11.0	0/208.0/11.0	0/195.5/10.0	1/983.5/51.5
FastText	0/189.0/10.0	0/168.0/7.0	1/149.5/6.0	0/200.0/9.0	0/153.0/7.0	1/859.5/39.0
EWE	1/102.5/3.0	0/206.0/11.0	0/144.0/4.0	0/165.5/7.0	0/160.5/8.0	1/778.5/33.0
GloveTW	1/96.0/2.0	0/202.0/10.0	1/152.0/7.0	0/109.5/5.0	1/123.0/3.0	3/682.5/27.0
SSWE	**5**/152.0/6.5	4/193.0/8.0	1/228.0/13.0	6/70.0/2.0	4/135.0/5.0	20/778.0/34.5
TF-IDF	3/166.0/8.0	0/212.0/12.0	1/243.0/14.0	4/106.5/4.0	3/211.5/11.0	11/939.0/49.0
DeepMoji	0/169.0/9.0	0/153.0/6.0	1/164.0/8.0	0/173.5/8.0	1/143.0/6.0	2/802.5/37.0
w2v-Araque	0/200.0/11.0	0/198.0/9.0	0/222.0/12.0	0/204.5/10.0	0/216.5/12.0	0/1041.0/54.0
w2v-Edin	0/221.0/12.0	2/89.0/2.0	2/95.0/2.0	0/105.0/3.0	2/110.0/2.0	6/620.0/**21.0**
Emo2Vec	4/117.5/5.0	4/143.5/5.0	2/191.0/10.0	**11/42.0/1.0**	**8/81.5/1.0**	**29/575.5**/22.0
BERT-static	0/261.0/13.0	0/111.0/3.0	3/112.5/3.0	0/234.0/12.0	0/217.5/13.0	3/936.0/44.0
RoBERTa-static	**5/82.0/1.0**	**11/56.0/1.0**	**8/75.5/1.0**	0/245.5/13.0	0/177.0/9.0	24/636.0/25.0
BERTweet-static	0/289.0/14.0	0/240.0/14.0	0/174.0/9.0	0/284.5/14.0	0/257.0/14.0	0/1244.5/65.0

Bold values indicate the results. Underline Underline values indicate top three best overall results in terms of total rank position

In the previous evaluations, we analyzed the predictive performance achieved by each representation for one classification algorithm at a time, focusing on the individual contribution of the text representations in the performance on the final task. Next, we investigate the classification performance of the final sentiment analysis process, that is, the combination of text representation and classifier. Considering that the final classification is a combination of both representation and classifier, an appropriate choice of the classification algorithm may affect the performance of a text representation. For this purpose, we present an overall evaluation of all possible combinations of text representations and classification algorithms, examining them as pairs {*text representation*, *classifier*}. More clearly, we evaluate the classification effectiveness of 70 possible combinations of text representations and classifiers (14×5) on the 22 datasets of tweets. Tables 7 and 8 present the top and the bottom ten results in terms of the average rank position, respectively. Specifically, for each dataset, we calculate a rank of the 70 combinations and then average the rank position of each combination over the 22 datasets.

From Table 7, we can note that the best overall results were achieved by using RoBERTa to train an SVM classifier for both accuracy and F_1-macro. Also, w2v-Edin + SVM and RoBERTA + MLP appear in the top three results along with RoBERTa + SVM. From Table 8, we can notice that the RF classifier often appears among the worst results.

Tables 9 and 10 show a summary of the results for each text representation and classifier, respectively, from best to worst, in terms of the average rank position. As we can observe, Emo2Vec, RoBERTa, and w2v-Edin appear in the top three, being the representations that achieved the best overall performances. Among the classifiers, we can note that SVM and MLP seem to be good choices in Twitter sentiment Analysis regarding the usage of static text representations. Conversely, RF achieved the worst overall performance across all evaluations.

Table 7 Top 10 results achieved by evaluating combinations of static word representation models and classifiers

Accuracy			F_1-macro		
Representation	Classifier	Avg. rank pos.	Representation	Classifier	Avg. rank pos.
RoBERTa-static	SVM	9.32	RoBERTa-static	SVM	8.59
RoBERTa-static	MLP	11.57	w2v-Edin	SVM	9.39
w2v-Edin	SVM	14.50	RoBERTa-static	MLP	12.52
w2v-Edin	MLP	15.36	BERT-static	SVM	14.70
BERT-static	MLP	16.68	w2v-GN	SVM	14.95
w2v-GN	SVM	17.80	w2v-Edin	MLP	15.55
BERT-static	SVM	19.02	RoBERTa-static	LR	16.02
Emo2Vec	XGB	21.23	BERT-static	MLP	17.48
w2v-GN	MLP	22.50	GloVe-TWT	LR	17.91
FastText	MLP	23.20	Emo2Vec	SVM	18.05

Table 8 Bottom 10 results achieved by evaluating combinations of static word representation models and classifiers

Accuracy			F_1-macro		
Representation	Classifier	Avg. rank pos.	Representation	Classifier	Avg. rank pos.
DeepMoji	RF	51.41	EWE	RF	56.86
BERTweet-static	XGB	52.14	DeepMoji	RF	57.43
FastText	RF	52.75	BERTweet-static	LR	57.45
GloVe-WP	RF	54.11	w2v-GN	RF	58.02
w2v-Araque	RF	54.68	FastText	RF	60.36
BERT-static	RF	57.18	w2v-Araque	RF	61.00
RoBERTa-static	RF	57.95	GloVe-WP	RF	61.16
BERT-static	LR	60.86	BERT-static	RF	63.91
BERTweet-static	RF	61.02	RoBERTa-static	RF	64.11
BERTweet-static	LR	66.09	BERTweet-static	RF	67.32

The top three static representations identified in the previous evaluation, i.e., RoBERTa, w2v-Edin, and Emo2Vec, are very different from each other. While w2v-Edin and Emo2Vec were trained from scratch on tweets, RoBERTa was trained on traditional English texts. However, among these, RoBERTa is the only Transformer-based model, which holds state-of-the-art performance in capturing context and semantics of terms from texts. Furthermore, regarding w2v-Edin, although it was trained with a more straightforward architecture (feedforward neural network) as compared to others, its training parameters were optimized for the emotion detection task on tweets (Bravo-Marquez et al. 2016), which may have helped determining the sentiment expressed in tweets.

Surprisingly, as shown in Table 9, BERTweet achieved the worst overall performance among all assessed text representations, despite having been trained using the same state-of-the-art Transformer-based architecture as RoBERTa *while yet using tweets*. One possible explanation for this behavior is that BERTweet training procedure limits the representation of its training tweets to 60 tokens only, while RoBERTa uses a limit of 512 tokens. For that reason, we believe that RoBERTa model is able to capture more semantic information to the tokens from its training vocabulary as compared to BERTweet when one collects the token representation from the embeddings layer.

In addition to the individual assessment of text representations and classifiers presented in Tables 9 and 10, Table 11 shows the best results achieved for each dataset. We can see that RoBERTa achieved the highest accuracies in seven out of the 22 datasets, and highest F_1-macro scores in nine out of the 22 datasets. Furthermore, as highlighted in Table 7, RoBERTA + SVM achieved the best performances in six out of the 22 datasets in terms of accuracy, and in eight out of the 22 datasets in terms of F_1-macro.

Finally, regarding research question RQ1, we can highlight and suggest that: (i) disregarding the classification algorithms, Emo2Vec, w2v-Edin, and RoBERTa seem to be well-suited representations for determining the sentiment expressed in tweets,

Table 9 Summary of the results for each static word representation model, from best to worst, in terms of the average rank position

Accuracy		F_1-macro	
Representation	Avg. rank pos.	Representation	Avg. rank pos.
Emo2Vec	26.35	Emo2Vec	25.34
RoBERTa-static	27.93	RoBERTa-static	29.06
w2v-Edin	29.55	w2v-Edin	30.05
w2v-GN	30.05	w2v-GN	31.04
GloVe-TWT	32.0	GloVe-TWT	31.55
EWE	34.5	SSWE	33.4
SSWE	34.78	EWE	34.15
DeepMoji	35.93	DeepMoji	34.98
TF-IDF	36.21	FastText	36.34
FastText	37.0	BERT-static	39.38
BERT-static	39.43	GloVe-WP	39.43
GloVe-WP	40.25	TF-IDF	40.25
w2v-Araque	43.15	w2v-Araque	42.85
BERTweet-static	49.88	BERTweet-static	49.18

Table 10 Summary of the results for each classifier, from best to worst, by evaluating the static word representations, in terms of the average rank position

Accuracy		F_1-macro	
Classifier	Avg. rank pos.	Classifier	Avg. rank pos.
MLP	26.28	SVM	23.07
SVM	28.3	MLP	26.69
XGB	35.83	LR	31.27
LR	39.17	XGB	41.33
RF	47.92	RF	55.15

and (ii) considering the combination of text representations and classifiers, RoBERTa + SVM achieved the best overall performance, which may represent a good choice for Twitter sentiment analysis in hardware-restricted environments, since the cost here is most due to the classifier induction.

5 Evaluation of the transformer-based text representations

In this section, we address the research question RQ2, as follows:

RQ2.Considering state-of-the-art Transformer-based autoencoder models, which are the most effective in the sentiment classification of tweets?

To answer that question, we conduct a thorough evaluation of the widely used BERT and RoBERTa models and the BERT-based transformer trained from scratch with tweets, namely, BERTweet. These models represent a set of the most recent

Table 11 Best results achieved for each dataset by evaluating the static word representation models

Dataset	Accuracy			F_1-macro		
	%	Classifier	Representation	%	Classifier	Representation
iro	78.81	LR	Emo2Vec	75.87	LR	Emo2Vec
sar	87.5	LR	SSWE	87.19	LR	SSWE
ntu	95.3	MLP	w2v-Edin	95.19	MLP	w2v-Edin
S15	90.35	LR	TF-IDF	78.48	SVM	RoBERTa-static
stm	87.74	SVM	w2v-Edin	87.67	SVM	w2v-Edin
per	83.83	MLP	w2v-GN	80.58	SVM	w2v-GN
hob	94.82	MLP	BERT-static	94.05	MLP	BERT-static
iph	84.39	MLP	GloVe-TWT	81.15	MLP	GloVe-TWT
mov	88.78	XGB	Emo2Vec	77.86	MLP	FastText
san	84.71	MLP	TF-IDF	84.56	MLP	TF-IDF
Nar	89.0	LR	SSWE	88.58	LR	SSWE
arc	87.6	MLP	RoBERTa-static	87.29	MLP	RoBERTa-static
S18	86.5	SVM	RoBERTa-static	86.4	SVM	RoBERTa-static
OMD	85.1	SVM	RoBERTa-static	83.85	SVM	RoBERTa-static
HCR	80.24	SVM	TF-IDF	74.55	SVM	RoBERTa-static
STS	89.08	LR	SSWE	87.7	LR	SSWE
SST	85.06	LR	Emo2Vec	84.77	LR	Emo2Vec
Tar	84.42	SVM	RoBERTa-static	84.42	SVM	RoBERTa-static
Vad	89.32	SVM	RoBERTa-static	87.8	SVM	RoBERTa-static
S13	88.24	XGB	Emo2Vec	85.59	LR	Emo2Vec
S17	89.03	SVM	RoBERTa-static	88.37	SVM	RoBERTa-static
S16	86.0	SVM	RoBERTa-static	83.18	SVM	RoBERTa-static

Transformer-based autoencoder language modeling techniques that have achieved state-of-the-art performance in many NLP tasks. While BERT is the first Transformer-based autoenconder model to appear in the literature, RoBERTa is an evolution of BERT with improved training methodology, due to the elimination of the Next Sentence Prediction task, which may fit NLP tasks on tweets considering they are limited in size and self-contained in context. Moreover, by evaluating BERTweet we analyze the performance of a Transformer-based model trained from scratch on tweets.

In this set of experiments, we give an example tweet as input to the transformer model and concatenate its last four layers to be the token representation and the tweet representation is the average of the tokens representation. Next, those representations collected from the whole dataset are given as input to the learning classifier method together with the labels of the tweets. Finally, the learned classifier is employed to perform the evaluation. In this way, we once again follow the feature extraction plus classification strategy but now using the contextualized embedding from each tweet.

Table 12 presents the classification results when using the SVM classifier in terms of accuracy and F_1-macro, and Table 13 shows a summary of the complete evaluation

Table 12 Accuracies and F_1-macro scores (%) achieved by evaluating the Transformer-based language models using the SVM classifier

Dataset	Accuracy			F_1-macro		
	RoBERTa	BERT	BERTweet	RoBERTa	BERT	BERTweet
iro	46.67	**71.43**	69.52	31.0	**66.51**	61.22
sar	57.86	**76.07**	61.96	46.23	**75.63**	59.24
ntu	78.45	87.02	**91.03**	78.18	86.73	**90.86**
S15	86.31	90.03	**91.59**	73.33	77.58	**82.06**
stm	84.12	89.14	**90.25**	84.04	89.11	**90.23**
per	72.66	83.13	**83.14**	71.18	80.73	**81.34**
hob	69.15	**83.52**	83.13	68.62	**81.76**	81.53
iph	75.96	81.58	**83.65**	74.65	79.63	**81.97**
mov	74.35	84.14	**86.47**	68.61	77.58	**80.55**
san	83.66	85.54	**89.87**	83.43	85.47	**89.81**
Nar	89.73	91.6	**95.35**	89.48	91.34	**95.22**
arc	88.18	87.25	**90.16**	88.0	87.02	**89.99**
S18	86.28	87.25	**88.97**	86.07	87.16	**88.87**
OMD	82.16	85.62	**87.36**	81.2	84.71	**86.4**
HCR	76.67	78.61	**79.82**	72.89	74.75	**76.22**
STS	89.48	90.46	**93.56**	88.11	89.16	**92.65**
SSt	84.01	85.19	**86.76**	83.83	84.87	**86.53**
Tar	84.83	85.64	**86.93**	84.81	85.62	**86.92**
Vad	87.73	89.63	**90.56**	86.24	88.18	**89.28**
S13	84.26	86.62	**88.15**	82.0	84.21	**86.13**
S17	90.61	91.54	**92.56**	90.08	91.03	**92.08**
S16	87.62	88.77	**90.72**	85.5	86.59	**88.86**
#wins	0	3	19	0	3	19
rank sums	65.0	42.0	25.0	65.0	42.0	25.0
position	3.0	2.0	1.0	3.0	2.0	1.0

{BERTweet} $\succ$ {RoBERTa, BERT}
{RoBERTa} $\succ$ {BERT}
Bold values indicate the best results

regarding all classifiers. As in previous section, to limit the number of tables in the manuscript, we only report the computational results in detail for the SVM classifier as an example of this evaluation. From Table 12, we can note that BERTweet achieved the best results in 18 out of the 22 datasets for both accuracy and F_1-macro. Precisely, the Friedman and the Nemenyi tests detected that BERTweet is significantly better than RoBERTa and BERT, while RoBERTa is better than BERT. Similarly, regarding all classifiers, Table 13 shows that BERTweet outperformed BERT and RoBERTa by a significant difference in terms of the total number of wins for both accuracy and F_1-macro.

Table 13 Overview of the results (number of wins, rank sum, and rank position, respectively) achieved by evaluating each Transformer-based model on the 22 datasets, for each classification algorithm

Model	LR	SVM	MLP	RF	XGB	Total
Accuracy						
BERT	2/45.0/2.0	0/65.0/3.0	3/47.0/2.0	4/46.5/2.0	3/43.5/2.0	12/247.0/11.0
RoBERTa	2/60.0/3.0	3/42.0/2.0	2/57.0/3.0	5/52.5/3.0	0/63.0/3.0	12/274.5/14.0
BERTweet	**18/27.0/1.0**	**19/25.0/1.0**	**17/28.0/1.0**	**15/33.0/1.0**	**20/ 25.5/1.0**	**89/138.5/5.0**
F_1-macro						
BERT	2/45.0/2.0	0/65.0/3.0	3/47.0/2.0	3/48.0/2.0	3/44.0/2.0	11/249.0/11.0
RoBERTa	2/60.0/3.0	3/42.0/2.0	2/57.0/3.0	5/52.5/3.0	0/61.0/3.0	12/272.5/14.0
BERTweet	**18/27.0/1.0**	**19/25.0/1.0**	**17/28.0/1.0**	**15/31.5/1.0**	**19/ 27.0/1.0**	**88/138.5/5.0**

Bold values indicate the best results

Table 14 Overall analysis of using the Transformer-based models to train each classification algorithm, examining them as pairs {language model, classifier}, in terms of the average rank position

Accuracy			F_1-macro		
Model	Classifier	Avg. rank pos.	Model	Classifier	Avg. rank pos.
BERTweet	LR	2.55	BERTweet	LR	2.32
BERTweet	MLP	2.68	BERTweet	MLP	3.0
BERTweet	SVM	4.16	BERTweet	SVM	3.55
RoBERTa	MLP	5.05	RoBERTa	LR	4.98
RoBERTa	LR	5.68	RoBERTa	MLP	5.39
BERT	MLP	6.23	BERT	SVM	5.75
BERT	SVM	6.86	BERT	MLP	6.61
BERTweet	XGB	7.34	BERT	LR	7.3
BERT	LR	7.73	BERTweet	XGB	8.36
RoBERTa	XGB	9.57	RoBERTa	XGB	10.18
BERT	XGB	11.61	RoBERTa	SVM	10.8
BERTweet	RF	11.95	BERT	XGB	11.77
RoBERTa	SVM	12.05	BERTweet	RF	12.48
RoBERTa	RF	12.98	RoBERTa	RF	13.55
BERT	RF	13.57	BERT	RF	13.98

Next, we present an overall analysis of using BERT, RoBERTa, and BERTweet models to train each one of the five classification algorithms, examining them as pairs {*language model*, *classifier*}. Table 14 presents the average rank position across all 15 possible combinations (3 language models $\times$ 5 classification algorithms), from best to worst, as explained in Sect. 4. We can observe that BERTweet combined with LR, MLP, and SVM classifiers achieved the best overall performances for both accuracy and F_1-macro. Conversely, using RF to train the Transformer-based embeddings seems to harm the performance of the models.

Table 15 Summary of the results for each Transformer-based model, from best to worst, in terms of the average rank position

Accuracy		F_1-macro	
Model	Avg. rank pos.	Model	Avg. rank pos.
BERTweet	5.74	BERTweet	5.94
RoBERTa	9.06	RoBERTa	8.98
BERT	9.20	BERT	9.08

Table 16 Summary of the results for each classifier, from best to worst, by evaluating the Transformer-based models, in terms of the average rank position

Accuracy		F_1-macro	
Classifier	Avg. rank pos.	Classifier	Avg. rank pos.
MLP	4.65	LR	4.86
LR	5.32	MLP	5.00
SVM	7.69	SVM	6.70
XGB	9.51	XGB	10.11
RF	12.83	RF	13.33

Tables 15 and 16 show a summary of the results for each model and classifier, respectively, from best to worst, in terms of the average rank position. From Table 15, we can see that BERTweet achieved the best overall classification effectiveness and was ranked first. Also, RoBERTa and BERT achieved comparable overall performances for both accuracy and F_1-macro. Regarding the classifiers, as shown in Table 16, MLP and LR achieved rather comparable performances and were ranked as the top two best classifiers regarding the Transformer-based models, followed by SVM, XGB, and RF.

Regarding the results achieved for each dataset, Table 17 presents the best results in terms of accuracy and F_1-macro. As we can notice, BERTweet outperformed BERT and RoBERTa in 17 out of the 22 datasets in terms of accuracy and in 18 out of the 22 datasets in terms of F_1-macro. These results may confirm that Twitter sentiment classification benefits most from contextualized language models trained from scratch on Twitter data. Unlike BERT and RoBERTa, which were trained on traditional English texts, BERTweet was trained on a huge amount of 850M tweets. This fact may have helped BERTweet on learning the specificities of tweets, such as their morphological and semantic characteristics.

For a better understanding of the results, we present an analysis of the difference between the vocabulary embedded in the assessed models. For this purpose, Table 18 highlights the number of tokens shared between BERT, RoBERTa, and BERTweet. In other words, we show the amount of tokens (in %) embedded in the models presented in each row that are also included in the models presented in each column, i.e., the intersection between their vocabularies. For example, regarding BERT (first row), we can see that 61% of its tokens can be found on RoBERTa (second column). The information below each model name in the columns refers to their vocabulary size (number of embedded tokens). It is possible to note that only 32% of the 64K tokens from BERTweet vocabulary (i.e., about 20K tokens) can be found in BERT. It means that, when compared to BERT, BERTweet contains about 44K (64 − 20) specific

Table 17 Best results achieved for each dataset by evaluating combinations of Transformer-based models and classifiers

Dataset	Accuracy			F_1-macro		
	%	Classifier	Model	%	Classifier	Model
iro	80.48	LR	BERT	73.08	LR	BERT
sar	76.07	SVM	BERT	75.63	SVM	BERT
ntu	91.03	LR	BERTweet	90.86	SVM	BERTweet
S15	92.23	LR	BERTweet	83.33	LR	BERTweet
stm	90.79	LR	RoBERTa	90.75	LR	RoBERTa
per	87.69	LR	BERTweet	85.29	LR	BERTweet
hob	87.93	LR	RoBERTa	86.29	LR	RoBERTa
iph	87.59	MLP	BERTweet	84.72	MLP	BERTweet
mov	89.47	MLP	RoBERTa	82.12	LR	BERTweet
sand	91.17	MLP	BERTweet	91.11	MLP	BERTweet
Nar	95.60	MLP	BERTweet	95.43	MLP	BERTweet
arc	90.74	MLP	BERTweet	90.51	MLP	BERTweet
S18	88.97	SVM	BERTweet	88.87	SVM	BERTweet
OMD	87.36	SVM	BERTweet	86.40	SVM	BERTweet
HCR	81.55	XGB	BERTweet	76.77	LR	BERTweet
STS	93.90	LR	BERTweet	92.98	LR	BERTweet
SSt	86.76	SVM	BERTweet	86.53	SVM	BERTweet
Tar	86.93	SVM	BERTweet	86.92	SVM	BERTweet
Vad	90.80	LR	BERTweet	89.38	LR	BERTweet
S13	89.61	LR	BERTweet	87.37	LR	BERTweet
S17	92.56	SVM	BERTweet	92.08	SVM	BERTweet
S16	91.03	LR	BERTweet	89.05	LR	BERTweet

Table 18 Percentage of vocabulary's tokens of the Transformer-based model in the row that are also in the vocabulary's tokens of the Transformer-based model in the column

	BERT $\|V\| = 30K$	RoBERTa $\|V\| = 50K$	BERTweet $\|V\| = 64K$
BERT	–	61	62
RoBERTa	41	–	71
BERTweet	32	55	–

tokens extracted from tweets. Similarly, 55% of the tokens embedded in BERTweet (i.e., about 35K tokens) can be found in RoBERTa, meaning that BERTweet holds about 29K (64 − 35) specific tokens from tweets that are not included in RoBERTa. As a matter of fact, analyzing the tokens embedded in BERTweet, we find some specific tokens, such as "Awww", "hahaha", "broo", and other internet expressions and slang that social media users often use to express themselves. While creating representations for these tokens is straightforward in BERTweet, BERT and RoBERTa need to do some extras steps. Specifically, when BERT and RoBERTa do not find a token in

Table 19 Top 10 results achieved for combinations of language model and classifier by evaluating the Transformer-based models and the static word representations, in terms of the average rank position

Accuracy			F_1-macro		
Model	Classifier	Avg. rank pos.	Model	Classifier	Avg. rank pos.
BERTweet	LR	6.20	BERTweet	LR	5.91
BERTweet	MLP	6.32	BERTweet	MLP	6.55
RoBERTa	MLP	10.27	BERTweet	SVM	9.14
BERT	MLP	10.43	BERT	SVM	10.00
BERTweet	SVM	10.45	RoBERTa	MLP	10.89
RoBERTa	LR	12.23	RoBERTa	LR	10.91
BERT	LR	12.91	BERT	MLP	10.91
BERT	SVM	13.11	BERT	LR	12.07
BERTweet	XGB	17.02	RoBERTa-static	SVM	17.32
RoBERTa-static	SVM	18.66	W2V-Edin	SVM	18.75

Table 20 Overall evaluation of the Transformer-based models and the static word representations, from best to worst, in terms of the average rank position

Accuracy		F_1-macro	
Model	Avg. rank pos.	Model	Avg. rank pos.
BERTweet	14.84	BERTweet	17.38
BERT	20.86	BERT	23.42
RoBERTa	23.00	RoBERTa	24.94
Emo2Vec	37.49	Emo2Vec	36.09
RoBERTa-static	39.65	RoBERTa-static	40.37
w2v-Edin	41.50	w2v-Edin	41.54
w2v-GN	42.37	w2v-GN	43.02
GloVe-TWT	44.64	GloVe-TWT	43.55
SSWE	46.75	SSWE	44.78
EWE	47.21	EWE	46.17
DeepMoji	48.35	DeepMoji	46.79
TF-IDF	49.09	FastText	48.95
FastText	50.18	GloVe-WP	51.55
BERT-static	52.30	BERT-static	51.67
GloVe-WP	53.01	TF-IDF	53.00
w2v-Araque	56.25	w2v-Araque	55.45
BERTweet-static	63.52	BERTweet-static	62.34

their vocabularies, they split the token into subtokens until all of them are found. For example, the token “hahaha” would be split into “ha”, “ha”, and “ha” to represent the original token. This analysis points out that this particular vocabulary, combined with a language model that was trained focused on learning the intrinsic structure of

tweets, is the responsible for the BERTweet language model's best performance on tweet sentiment classification.

In this context, regarding RQ2, we believe BERTweet is an effective language modeling technique in distinguishing the sentiment expressed in tweets. Also, regarding the classifiers, in general, MLP and LR seem to be good choices when using Transformer-based models.

Different from static representations, when we used only the embedding layer of the language models, in this section, we use the whole language model: the tweet goes from the embedding layer up to the last layer to be transformed into a vector representation. Attempting to understand the benefits from using the whole language model (embedding layer and language model), we compare the predictive performance of Transformer-based models evaluated in this section against all the static representations assessed in Sect. 4. Table 19 presents the top ten results across all 85 possible combinations of models and classifiers (17 models × 5 classification algorithms), and Table 20 shows an overall evaluation of the models, from best to worst, in terms of the

Table 21 Best results achieved for each dataset by evaluating combinations of language models and classifiers, regarding Transformer-based models and static word representations

Dataset	Accuracy			F_1-macro		
	%	Classifier	Model	%	Classifier	Model
iro	80.48	LR	BERT	75.87	LR	Emo2Vec
sar	87.50	LR	SSWE	87.19	LR	SSWE
ntu	95.30	MLP	w2v-Edin	95.19	MLP	w2v-Edin
S15	92.23	LR	BERTweet	83.33	LR	BERTweet
stm	90.79	LR	RoBERTa	90.75	LR	RoBERTa
per	87.69	LR	BERTweet	85.29	LR	BERTweet
hob	94.82	MLP	BERT-static	94.05	MLP	BERT-static
iph	87.59	MLP	BERTweet	84.72	MLP	BERTweet
mov	89.47	MLP	RoBERTa	82.12	LR	BERTweet
san	91.17	MLP	BERTweet	91.11	MLP	BERTweet
Nar	95.60	MLP	BERTweet	95.43	MLP	BERTweet
arc	90.74	MLP	BERTweet	90.51	MLP	BERTweet
S18	88.97	SVM	BERTweet	88.87	SVM	BERTweet
OMD	87.36	SVM	BERTweet	86.40	SVM	BERTweet
HCR	81.55	XGB	BERTweet	76.77	LR	BERTweet
STS	93.90	LR	BERTweet	92.98	LR	BERTweet
SST	86.76	SVM	BERTweet	86.53	SVM	BERTweet
Tar	86.93	SVM	BERTweet	86.92	SVM	BERTweet
Vad	90.80	LR	BERTweet	89.38	LR	BERTweet
S13	89.61	LR	BERTweet	87.37	LR	BERTweet
S17	92.56	SVM	BERTweet	92.08	SVM	BERTweet
S16	91.03	LR	BERTweet	89.05	LR	BERTweet

average rank position. In addition, Table 21 shows the best results achieved for each dataset.

From Tables 19 and 20 we can notice that the Transformer-based BERTweet model outperformed all other models and was ranked first in both evaluations. Also, Table 20 shows that the Transformed-based models achieved the best overall results against all static models and were ranked as the top three best representations. Furthermore, from Table 21, the Transformer-based BERTweet model achieved the best overall classification effectiveness in 16 out of the 22 datasets in terms of accuracy and in 17 out of the 22 datasets in terms of F_1-macro.

These results point out that learning language model parameters is essential in distinguishing the sentiment expressed in tweets. Static representations may lose a lot of relevant information considering they ignore the diversity of meaning that words may have depending on the context they appear. In contrast, Transformer-based models benefit from learning how to encode the context information of a token in an embedding.

6 Adapting transformer-based models to a large collection of English tweets

In this section, we aim at performing computational experiments in order to answer the research question RQ3, stated as follows:

RQ3. Can a second phase of adaptive pretraining of Transformer-based autoencoder models using a large set of English tweets improve the sentiment classification performance?

To answer this research question, we evaluate the classification effectiveness of BERT, RoBERTa, and BERTweet language models adapted with tweets from a corpus of 6.7M unlabeled, or generic unlabeled, tweets, as described in Sect. 3.3. Precisely, we use this set of tweets to adapt the model weights using the intermediate masked language model task as the training objective with the probability of 15% to (randomly) mask tokens in the input. We also compare the results of such adapted models against those achieved by using the original weights of the Transformer-based models, as presented in Sect. 5, in order to analyze whether the adjustment of the models via a second phase of pretraining improves the predictive performance of the sentiment classification.

In general, the performance of the adapted models is very sensitive to different random seeds (Dodge et al. 2020). For that reason, all the results presented in this section are the average of three executions using different seeds (12,34,56) to account for the sensitivity of the adaptive process regarding different seeds.

The first part of the experiments reported in this section consists in determining whether the predictive performance of the Transformer-based models are affected by the adaptation procedure using tweets from corpora of different sizes. For this purpose, in addition to the entire Edinburgh corpus of 6,657,700 tweets (around 6.7M tweets), we used nine other smaller samples of tweets with different sizes, varying from 500 to 1.5M tweets. Specifically, we generated samples containing 0.5K, 1K, 5K, 10K, 25K, 50K, 250K, 500K, and 1.5M generic unlabeled tweets. In the adaptation processes,

we performed three training epochs, except for the adaptation of models with 6.7M tweets, when we used one epoch, as there was a degradation of some models, such as BERTweet. In all adaptation process, all layers are unfrozen. Regarding the batch size, we use the available hardware capacity of eight instances per device. We used a learning rate of 5e-5 with a linear scheduler and Adam optmizer with beta1 equal to 0.9, beta2, 0.999, and epsilon, 1e-8. We also use a max gradient of 1 and with no weight decay.

Tables 22, 23, and 24 present the average classification accuracies and F_1-macro scores when adapting BERT, RoBERTa, and BERTweet, respectively, with the different samples of tweets generated from the Edinburgh corpus. As in previous sections, for space constraints, we only report the detailed evaluation using the SVM classifier. Regarding the variance in performance across the different seeds, the mean and maximum standard deviations are 0.05% and 0.5% for both accuracy and F_1-macro, respectively.

Note that BERT (Table 22) was benefited most when adapted with the sample of 250K tweets, being ranked first in the overall evaluation (position row), for both accuracy and F_1-macro. Although these results are only significantly better than those by adapting BERT with 6.7M tweets, they may be a piece of evidence that more tweets does not necessarily means better performance for adapted models. RoBERTa (Table 23) achieved the best overall results when adapted with the sample of 1.5M, for both accuracy and F_1-macro. However, these results are only significantly better than those by adapting RoBERTa with the sample of 0.5K. On the other hand, BERTweet (Table 24) benefited from smaller samples, achieving higher overall predictive performances when adapted with the sample of 25K, for both accuracy and F_1-macro, being significantly better than the results achieved with samples of 0.5K, 1.5M, and 6.7M. This is an expected result as BERTweet is already trained from scratch from tweets. As we are adapting the language model, BERT and RoBERTa seems to require more samples to accommodate the Twitter-based vocabulary into the weights' model.

Next, we analyze the overall performance of the adapted Transformer-based models for each classification algorithm. Tables 25 and 26 summarize the results in terms of accuracy and F_1-macro, respectively. Regarding the variance across the different seeds, the mean and maximum standard deviations are 0.2% and 0.7% in terms of accuracy, and 0.26% and 0.98% in terms of F_1-macro.

Interestingly, from Tables 25 and 26, we can note that when adapting a language model to fit a specific type of text, such as tweets, applying large corpora does not guarantee better predictive performances. Specifically, the best overall results (Total column) were achieved when adapting BERT, RoBERTa, and BERTweet models with samples of 250K, 50K, and 5K tweets, respectively, for both accuracy and F_1-macro.

Regarding the results achieved for each dataset, Table 27 shows the best predictive performances in terms of accuracy and F_1-macro. We can see that BERTweet achieved the best results for most datasets when the adaptive pretraining employs a fewer number of tweets. More specifically, BERTweet outperformed the other models when adapted with samples varying from 1K to 25K tweets in 14 out of the 22 datasets for both accuracy and F_1-macro.

As in previous sections, we also present an overall evaluation of combining all adapted models and classifiers across the 22 datasets, in terms of the average rank

Table 22 Average classification accuracies and F_1-macro scores (%) achieved by adaptive pretraining of BERT with different samples of generic unlabeled tweets, using the SVM classifier

Dataset	0.5K	1K	5K	10K	25K	50K	250K	500K	1.5M	6.7M
Accuracy										
Iro	73.81	74.05	72.38	66.19	76.43	73.57	**76.67**	61.19	62.07	56.74
Sar	68.93	70.18	74.46	67.32	73.04	**77.32**	74.46	70.36	71.31	63.69
Ntu	88.84	86.65	84.18	85.97	83.08	85.62	84.89	92.09	93.88	**94.38**
S15	90.35	89.4	90.02	**90.64**	88.78	90.03	87.22	87.23	85.96	87.13
Stm	89.14	89.14	88.29	**89.41**	89.13	88.86	89.13	88.3	88.57	87.01
Per	84.72	82.91	**85.42**	83.14	82.91	82.01	81.1	81.32	79.5	76.16
Hob	83.9	85.24	86.39	**86.78**	86.18	84.85	84.29	83.72	84.15	80.89
Iph	81.02	81.39	81.58	82.15	81.02	81.2	82.15	81.21	80.34	78.96
Mov	82.72	83.43	84.49	85.2	84.5	85.03	**85.91**	83.26	82.41	82.71
San	86.19	86.19	85.87	87.01	86.27	87.42	**87.66**	87.0	86.98	87.6
Nar	91.93	91.44	92.42	91.52	91.6	91.28	92.5	92.83	**93.67**	93.5
Arc	86.85	88.07	88.94	88.19	87.49	88.01	87.95	**89.58**	88.13	88.5
S18	**87.25**	86.82	86.45	86.45	86.5	86.61	86.5	86.82	87.09	86.0
OMD	85.68	**85.99**	85.26	85.15	84.73	84.31	85.79	84.84	85.43	84.58
HCR	78.56	78.82	78.4	78.35	78.35	**79.5**	78.82	77.98	77.32	76.56
STS	90.36	90.51	90.41	90.81	90.22	91.05	91.79	**92.23**	91.89	91.29
SSt	84.58	85.06	**85.37**	84.58	84.01	83.79	84.97	84.8	84.84	84.43
Tar	85.67	85.67	85.93	**86.01**	85.9	85.84	85.9	85.69	85.69	85.4
Vad	89.75	89.66	89.63	89.44	89.56	90.3	90.28	89.51	**90.5**	90.0
S13	86.52	86.66	**87.64**	87.05	87.39	86.96	87.23	86.98	87.07	86.32
S17	91.08	91.11	91.05	**91.15**	90.61	90.92	90.63	90.18	89.85	89.56
S16	88.45	88.34	**89.24**	88.83	88.83	88.82	88.9	88.29	88.01	87.93
#wins	1	1	4	**5**	0	2	3	2	2	1
Rank sums	125.5	112.0	101.0	101.5	132.5	117.0	**88.0**	132.0	129.5	171.0
Position	6.0	4.0	2.0	3.0	9.0	5.0	**1.0**	8.0	7.0	10.0
F_1-macro										
Iro	63.97	62.48	60.67	61.05	72.69	67.69	**73.0**	57.99	56.87	52.33
Sar	67.75	68.06	73.18	65.52	71.76	**75.85**	71.69	68.09	70.25	57.63
Ntu	88.5	86.2	83.84	85.64	82.51	85.44	84.69	91.89	93.69	**94.23**
S15	78.55	77.78	78.01	**78.72**	76.38	78.55	74.01	74.55	72.5	77.78
Stm	89.12	89.12	88.26	**89.4**	89.1	88.8	89.11	88.24	88.5	86.93
Per	82.78	80.57	**83.3**	80.77	80.39	79.59	78.69	79.28	77.47	74.41
Hob	82.21	83.57	84.87	**85.22**	84.89	83.11	82.78	82.33	83.06	79.7
Iph	79.28	79.56	79.6	80.16	79.2	79.02	**80.27**	79.42	78.52	77.48
Mov	75.81	76.51	78.12	77.69	77.73	77.92	**79.22**	76.36	75.66	76.16
San	86.13	86.1	85.81	86.96	86.22	87.36	**87.57**	86.93	86.9	87.51
Nar	91.67	91.17	92.18	91.27	91.35	91.01	92.28	92.59	**93.47**	93.31
Arc	86.62	87.86	88.72	87.97	87.26	87.77	87.72	**89.35**	87.91	88.33

Table 22 continued

Dataset	0.5K	1K	5K	10K	25K	50K	250K	500K	1.5M	6.7M
S18	**87.15**	86.71	86.33	86.36	86.38	86.48	86.37	86.7	86.97	85.82
OMD	84.72	**85.1**	84.28	84.18	83.72	83.42	84.81	83.84	84.47	83.63
HCR	74.84	75.06	74.66	74.58	74.54	**75.78**	75.05	74.23	73.4	72.72
STS	89.04	89.21	89.14	89.51	88.88	89.83	90.6	**91.1**	90.72	90.08
SSt	84.22	84.74	**85.05**	84.26	83.7	83.49	84.69	84.5	84.59	84.15
Tar	85.65	85.66	85.92	**86.0**	85.89	85.82	85.88	85.68	85.68	85.38
Vad	88.4	88.25	88.13	87.92	88.06	88.92	88.96	88.13	**89.16**	88.63
S13	84.14	84.25	**85.24**	84.61	85.12	84.67	84.9	84.6	84.83	84.08
S17	90.57	90.59	90.54	**90.62**	90.09	90.38	90.08	89.62	89.26	88.99
S16	86.23	86.14	**87.08**	86.6	86.63	86.6	86.74	86.07	85.78	85.72
#wins	1	1	4	**5**	0	2	4	2	2	1
Rank sums	127.0	113.0	99.5	104.5	131.0	115.0	**92.0**	132.0	129.5	166.5
Position	6.0	4.0	2.0	3.0	8.0	5.0	**1.0**	9.0	7.0	10.0

Accuracy: {5K, 10K, 250K} $\succ$ {6.7M}
F_1-macro: {5K, 250K} $\succ$ {6.7M}
Bold values indicate the best results

Table 23 Average classification accuracies and F_1-macro scores (%) achieved by adaptive pretraining of RoBERTa with different samples of generic unlabeled tweets, using the SVM classifier

Dataset	0.5K	1K	5K	10K	25K	50K	250K	500K	1.5M	6.7M
Accuracy										
Iro	46.67	46.67	46.67	**48.1**	46.67	46.67	46.67	46.67	45.24	46.74
Sar	60.71	59.29	65.0	57.86	62.14	60.71	65.0	62.14	63.57	64.7
Ntu	81.69	81.31	79.15	81.34	82.06	85.63	87.08	90.67	**91.75**	90.41
S15	83.18	83.5	84.75	86.93	84.75	85.38	86.93	**88.79**	87.24	87.03
Stm	85.23	86.35	89.7	85.24	87.75	88.04	88.03	87.48	**90.81**	86.35
Per	69.92	69.68	70.38	68.33	71.74	71.28	72.66	74.26	76.32	**77.0**
Hob	73.76	71.44	73.56	74.32	72.41	77.4	77.38	77.2	**77.97**	77.82
Iph	78.41	78.41	77.46	78.96	78.78	79.71	79.89	**80.09**	79.15	79.02
Mov	71.14	76.84	**81.64**	80.76	78.44	78.09	79.52	80.4	81.12	81.05
San	84.39	84.23	85.21	85.78	85.13	86.27	85.86	87.17	**88.23**	86.16
Nar	91.04	92.18	92.58	92.25	91.2	92.58	92.74	92.83	93.07	**93.88**
Arc	88.88	88.53	89.0	87.72	88.48	89.23	88.59	**89.64**	89.47	88.57
S18	**87.9**	87.15	87.15	87.09	86.61	87.52	87.14	87.47	86.61	86.62
OMD	82.79	83.58	83.63	**84.0**	83.53	83.47	83.79	82.27	82.74	82.55
HCR	76.52	77.25	77.2	76.73	76.1	**77.78**	76.41	77.77	75.94	77.74
STS	89.62	90.71	91.39	90.95	91.84	92.08	92.08	92.52	**92.92**	92.0
SSt	84.67	**86.28**	85.93	85.98	85.67	86.06	85.8	86.02	86.02	85.22
Tar	85.43	85.67	**86.36**	85.78	85.95	85.41	85.52	85.23	85.98	84.33
Vad	88.25	89.56	89.82	89.73	89.3	89.63	89.85	90.4	**91.11**	89.87

Table 23 continued

Dataset	0.5K	1K	5K	10K	25K	50K	250K	500K	1.5M	6.7M
S13	85.59	85.54	86.18	86.57	85.68	85.93	86.23	85.54	**87.03**	85.89
S17	90.77	91.0	**91.37**	91.29	91.07	91.15	91.02	90.86	90.63	89.68
S16	88.37	88.4	88.85	89.01	88.96	88.26	88.74	88.96	89.01	87.95
#wins	1	1	3	2	0	1	0	3	**7**	2
Rank sums	174.0	161.0	111.0	124.0	148.0	105.5	101.5	92.0	**78.5**	114.5
Position	10.0	9.0	5.0	7.0	8.0	4.0	3.0	2.0	**1.0**	6.0
F_1-macro										
Iro	31.0	31.0	31.0	34.71	31.0	31.0	31.0	31.0	32.8	**40.15**
Sar	48.82	48.19	55.76	48.13	51.35	49.02	56.14	54.16	55.33	**62.88**
Ntu	81.4	80.78	78.72	80.73	81.56	85.3	86.71	90.45	**91.49**	90.2
S15	69.86	69.79	71.77	74.52	71.6	73.14	75.94	**77.93**	74.99	74.64
Stm	85.2	86.3	89.68	85.23	87.71	88.01	87.95	87.45	**90.8**	86.29
Per	68.73	68.78	69.53	67.39	70.46	70.22	71.47	73.03	74.56	**75.72**
Hob	72.9	70.88	72.58	73.58	71.59	76.21	76.46	75.94	**76.62**	76.34
Iph	77.11	77.09	75.99	77.36	77.52	78.38	78.5	**78.72**	77.93	77.66
Mov	65.1	71.47	**75.5**	74.47	72.33	72.09	73.4	74.05	74.71	73.74
San	84.25	84.07	85.05	85.65	85.02	86.11	85.73	87.02	**88.11**	86.03
Nar	90.8	91.96	92.38	92.03	90.97	92.39	92.56	92.64	92.9	**93.72**
Arc	88.68	88.33	88.81	87.52	88.28	89.04	88.42	**89.45**	89.28	88.38
S18	**87.76**	87.02	87.0	86.93	86.46	87.37	87.01	87.35	86.47	86.51
OMD	81.79	82.72	82.6	**83.09**	82.49	82.47	82.77	81.34	81.69	81.5
HCR	73.02	73.58	73.24	72.91	72.23	**74.05**	72.52	74.04	71.92	73.99
STS	88.37	89.55	90.26	89.8	90.71	90.97	91.02	91.41	**91.91**	90.87
SSt	84.43	**86.05**	85.69	85.69	85.42	85.86	85.55	85.83	85.8	84.98
Tar	85.41	85.65	**86.34**	85.76	85.94	85.39	85.5	85.21	85.97	84.31
Vad	86.81	88.22	88.49	88.38	87.91	88.38	88.49	89.16	**89.93**	88.51
S13	83.42	83.31	84.04	84.48	83.46	83.83	84.03	83.47	**84.94**	83.74
S17	90.24	90.5	**90.87**	90.77	90.57	90.64	90.51	90.34	90.09	89.09
S16	86.27	86.28	86.82	86.96	86.86	86.16	86.64	86.94	**86.97**	85.76
#wins	1	1	3	1	0	1	0	3	**8**	4
Rank sums	175.0	160.0	114.0	129.0	151.0	106.5	98.5	88.0	**72.0**	116.0
Position	10.0	9.0	5.0	7.0	8.0	4.0	3.0	2.0	**1.0**	6.0

{50K, 250K, 500K, 1.5M} $\succ$ {0.5K}
{25K} $\succ$ {1.5M}
{1K} $\succ$ {0.5K, 1.5M}
Bold values indicate the best results

position. Table 28 shows the top ten results among all 150 possible combinations (3 models $\times$ 10 samples of tweets $\times$ 5 classification algorithms). As we can see in Table 28, adapted BERTweet embeddings achieved the best overall performances when used to train LR, MLP, and SVM, mastering the top ten results. Also, note that

Table 24 Average classification accuracies and F_1-macro scores (%) achieved by adaptive pretraining of BERTweet with different samples of generic unlabeled tweets, using the SVM classifier

Dataset	0.5K	1K	5K	10K	25K	50K	250K	500K	1.5M	6.7M
Accuracy										
Iro	68.81	68.1	71.19	73.81	74.76	**79.76**	71.67	70.48	67.78	66.43
Sar	64.82	69.11	67.68	73.21	73.21	68.93	70.36	70.36	72.26	71.79
Ntu	89.95	91.4	91.03	92.82	93.17	92.09	93.16	93.16	**94.38**	92.45
S15	90.34	92.22	90.04	**92.53**	92.23	91.6	90.03	90.04	89.4	89.09
Stm	90.52	91.92	**93.04**	91.36	91.07	91.9	92.19	93.02	91.45	90.52
Per	84.28	84.04	85.64	86.77	**86.79**	85.41	86.1	85.87	83.14	81.32
Hob	85.06	85.25	86.78	86.78	86.78	**87.92**	86.6	86.77	87.03	86.21
Iph	84.04	83.84	83.47	85.35	84.78	84.22	83.47	**85.55**	82.22	82.34
Mov	85.93	87.71	88.6	88.95	88.42	**89.49**	88.78	88.42	87.3	87.89
San	90.03	90.69	91.42	90.93	91.18	**91.67**	91.01	91.01	89.46	90.03
Nar	95.93	96.33	96.58	96.41	96.58	96.41	96.58	95.92	95.71	94.87
Arc	90.05	90.39	90.63	90.34	91.09	90.45	**91.21**	90.98	91.12	90.74
S18	89.99	89.78	90.26	**90.48**	89.89	89.83	89.94	89.03	88.26	87.9
OMD	88.09	88.93	**88.99**	87.67	88.25	88.51	88.88	88.09	87.84	86.88
HCR	80.24	80.5	81.23	81.18	**81.24**	80.6	80.61	78.98	78.25	78.09
STS	94.35	**95.23**	95.08	94.99	94.84	95.13	94.49	94.15	93.97	94.3
SSt	87.64	88.16	88.73	**89.6**	88.99	88.03	87.51	87.59	86.98	87.68
Tar	87.02	**87.83**	87.68	87.54	87.68	87.6	87.28	86.79	86.38	85.84
Vad	90.8	92.42	92.56	92.66	92.23	92.59	**92.71**	92.18	92.11	92.09
S13	89.4	90.0	**90.25**	89.84	90.0	88.88	89.38	88.85	88.76	88.44
S17	92.75	93.37	93.35	**93.49**	93.4	92.97	92.75	92.01	91.31	91.16
S16	90.7	91.38	**91.98**	91.35	91.54	91.36	91.18	90.6	90.38	89.55
#wins	0	2	**4**	**4**	2	**4**	2	1	1	0
Rank sums	165.0	116.5	84.5	82.0	**70.0**	94.5	103.5	135.0	169.0	190.0
Position	8.0	6.0	3.0	2.0	**1.0**	4.0	5.0	7.0	9.0	10.0
F_1-macro										
Iro	52.83	60.79	60.98	68.12	67.33	**73.81**	66.15	65.03	59.86	56.98
Sar	59.88	67.4	65.46	71.66	**71.79**	65.96	66.98	67.17	70.79	68.66
Ntu	89.77	91.23	90.85	92.69	93.0	91.95	93.01	93.02	**94.26**	92.3
S15	79.96	83.6	79.65	**84.24**	83.77	81.34	77.51	77.31	76.28	75.45
Stm	90.49	91.89	**93.02**	91.33	91.04	91.86	92.17	92.99	91.41	90.46
Per	82.51	82.16	83.63	**84.86**	84.83	83.3	83.97	83.86	81.12	79.22
Hob	83.54	83.59	85.3	85.27	85.36	**86.59**	85.0	85.09	85.29	84.63
Iph	82.45	82.18	81.7	83.75	82.88	82.46	81.64	83.75	80.39	80.6
Mov	79.52	82.05	83.35	83.48	82.66	**84.0**	82.67	82.06	79.87	80.2
San	89.96	90.6	91.35	90.84	91.09	**91.61**	90.94	90.94	89.35	89.91
Nar	95.78	96.2	96.45	96.28	96.44	96.27	96.45	95.78	95.55	94.69

Table 24 continued

Dataset	0.5K	1K	5K	10K	25K	50K	250K	500K	1.5M	6.7M
arc	89.88	90.24	90.46	90.16	90.91	90.28	**91.04**	90.8	90.94	90.56
S18	89.91	89.69	90.18	**90.4**	89.81	89.75	89.86	88.94	88.15	87.81
OMD	87.27	88.05	**88.17**	86.75	87.36	87.58	88.04	87.16	86.89	85.89
HCR	76.86	77.03	**78.01**	77.69	77.84	77.09	77.09	75.31	74.32	74.23
STS	93.51	**94.52**	94.36	94.23	94.08	94.38	93.66	93.3	93.02	93.42
SSt	87.39	87.92	88.49	**89.36**	88.75	87.78	87.23	87.35	86.73	87.45
Tar	87.01	**87.82**	87.68	87.53	87.68	87.59	87.28	86.78	86.37	85.83
Vad	89.54	91.33	91.5	91.6	91.13	91.48	91.6	91.06	90.98	90.96
S13	87.44	88.07	**88.34**	87.83	88.03	86.81	87.35	86.71	86.65	86.25
S17	92.29	92.94	92.92	**93.07**	92.97	92.52	92.28	91.5	90.77	90.62
S16	88.83	89.63	**90.3**	89.57	89.75	89.54	89.34	88.67	88.41	87.46
#wins	0	2	6	**7**	1	4	3	1	1	0
Rank sums	166.5	112.0	80.0	79.0	**73.5**	96.5	106.0	135.5	170.0	191.0
Position	8.0	6.0	3.0	2.0	**1.0**	4.0	5.0	7.0	9.0	10.0

$\{5K, 10K, 25K\} \succ \{0.5K, 1.5M, 6.7M\}$
Bold values indicate the best results

by using LR, MLP, and SVM, BERTweet outperformed all other models when adapted with samples containing 50K tweets or less.

Tables 29 and 30 show the top ten results among all adapted models and a summary of the results for each classifier, from best to worst, respectively, in terms of the average rank position. From Table 29, we can notice that all BERTweet adapted models (0.5K, 1K, 5K, 10K, 25K, 50K, 250K, 500K, 1.5M, and 6.7M) were ranked in the top ten results. Furthermore, neither BERT nor RoBERTa appear in the top results, even when they are adapted with the entire corpus of 6.7M tweets. RoBERTa appears only in the top 24 accuracy score with an average rank of 37.02 tuned with 50K tweets and combined MLP classifier and in top 28 F_1-macro score with an average rank of 37.27 tuned with 50K tweets and combined LR classifier. BERT appears only in the top 56 accuracy score with an average rank of 66.05 tuned with 1.5M tweets and combined MLP classifier and in top 51 F_1-macro score with an average rank of 60.77 tuned with 6.7M tweets and combined LR classifier. Among the classifiers, as we can see in Table 30, MLP and LR achieved the best predictive performances and were ranked as the top two best classifiers. Conversely, RF was ranked as the worst classifier.

From all previous evaluations, we can note that as the size of the samples increases, the adaptation procedure seems to be less effective. It may be due to the adjustment of the weights of the models' layers during the back-propagation process. Considering that the adaptation procedure consists in unfreezing the entire model obtained previously and adjusting their weights with the new data, the original model and the semantic and syntactic knowledge learned in its layers are changed. In that case, we believe that after some training iterations, the adjustment of the weights starts to damage the original knowledge embedded in the models' layers. The aforementioned conclusion may further explain why BERTweet achieved improved classification per-

Table 25 Overview of the results (number of wins, rank sum, and rank position, respectively) achieved by each classifier when adapting the Transformer-based models with different samples of unlabeled tweets in terms of accuracy

Sample	LR	SVM	MLP	RF	XGB	Total
BERT						
0.5k	3/127.5/5.5	1/125.5/6.0	1/129.0/7.0	1/153.0/10.0	**5**/108.0/2.0	11/643.0/30.5
1k	0/134.0/7.0	1/112.0/4.0	1/139.5/8.0	3/114.0/3.5	2/113.5/3.0	7/613.0/25.5
5k	3/115.0/4.0	4/101.0/2.0	3/114.5/5.0	1/120.5/6.0	1/128.0/8.0	12/579.0/25.0
10k	0/143.5/9.0	**5**/101.5/3.0	1/142.0/10.0	**4**/119.0/5.0	2/125.5/7.0	12/631.5/34.0
25k	0/136.0/8.0	0/132.5/9.0	0/141.0/9.0	1/134.0/8.0	2/146.0/10.0	3/689.5/44.0
50k	0/146.0/10.0	2/117.0/5.0	2/121.5/6.0	3/113.5/2.0	0/131.5/9.0	7/629.5/32.0
250k	0/127.5/5.5	3/**88.0/1.0**	3/**101.5/1.0**	2/**73.5/1.0**	2/**97.5/1.0**	10/**488.0/9.5**
500k	1/110.5/3.0	2/132.0/8.0	2/108.0/4.0	1/114.0/3.5	2/119.5/5.0	8/584.0/23.5
1.5M	4/96.0/2.0	2/129.5/7.0	3/107.0/3.0	1/131.5/7.0	1/122.0/6.0	11/586.0/25.0
6.7M	**10/74.0/1.0**	1/171.0/10.0	**6**/106.0/2.0	**4**/137.0/9.0	**5**/118.5/4.0	**26**/606.5/26.0
RoBERTa						
0.5k	1/140.0/9.0	1/174.0/10.0	0/165.5/9.0	0/171.5/9.0	0/173.0/10.0	2/824.0/47.0
1k	2/137.0/8.0	1/161.0/9.0	2/143.0/8.0	0/165.0/8.0	2/130.5/7.0	7/736.5/40.0
5k	3/**92.0/1.0**	3/111.0/5.0	0/99.5/3.5	4/104.0/5.0	4/100.0/**4.0**	14/506.5/18.5
10k	1/125.0/7.0	2/124.0/7.0	**4**/111.5/6.0	1/120.0/7.0	3/118.5/6.0	11/599.0/33.0
25k	**4**/103.0/3.0	0/148.0/8.0	2/107.5/5.0	1/104.5/6.0	2/98.0/3.0	9/561.0/25.0
50k	**4**/100.5/2.0	1/105.5/4.0	3/**85.5/1.0**	4/85.0/2.0	1/97.0/2.0	13/**473.5/11.0**
250k	0/124.0/6.0	0/101.5/3.0	3/131.5/7.0	**8/77.5/1.0**	**4/88.5/1.0**	15/523.0/18.0
500k	3/113.5/5.0	3/92.0/2.0	2/99.5/3.5	3/100.5/4.0	2/109.5/5.0	13/515.0/19.5
1.5M	3/109.0/4.0	**7/78.5/1.0**	3/91.5/2.0	1/98.5/3.0	2/136.0/8.0	**16**/513.5/18.0
6.7M	0/166.0/10.0	2/114.5/6.0	2/175.0/10.0	0/183.5/10.0	1/159.0/9.0	5/798.0/45.0
BERTweet						
0.5k	1/143.0/7.0	0/165.0/8.0	0/167.0/9.0	0/174.5/8.0	0/152.5/8.0	1/802.0/40.0
1k	**5**/78.5/2.0	2/116.5/6.0	3/95.0/4.0	0/99.0/4.0	4/76.5/2.0	14/465.5/18.0
5k	**5/69.5/1.0**	**4**/84.5/3.0	3/**75.5/1.0**	**10/42.5/1.0**	**10/56.0/1.0**	**32/328.0/7.0**
10k	2/92.0/3.0	**4**/82.0/2.0	4/95.5/5.0	2/72.5/3.0	4/81.0/3.0	16/423.0/16.0
25k	1/95.0/4.0	2/**70.0/1.0**	3/78.5/2.0	5/71.0/2.0	0/112.0/4.0	11/426.5/13.0
50k	2/110.0/5.0	4/94.5/4.0	**6**/91.0/3.0	2/117.0/5.5	3/119.5/6.0	17/532.0/23.5
250k	2/114.5/6.0	2/103.5/5.0	0/128.0/6.0	0/117.0/5.5	0/138.0/7.0	4/601.0/29.5
500k	0/162.0/8.0	1/135.0/7.0	0/150.0/7.0	2/138.0/7.0	1/117.0/5.0	4/702.0/34.0
1.5M	0/172.5/9.0	1/169.0/9.0	0/174.0/10.0	0/176.0/9.0	0/171.0/9.0	1/862.5/46.0
6.7M	1/173.0/10.0	0/190.0/10.0	3/155.5/8.0	0/202.5/10.0	0/186.5/10.0	4/907.5/48.0

Bold values indicate the best results

Table 26 Overview of the results (number of wins, rank sum, and rank position, respectively) achieved by each classifier when adapting the Transformer-based models with different samples of unlabeled tweets in terms of F_1-macro

Sample	LR	SVM	MLP	RF	XGB	Total
BERT						
0.5k	3/128.0/6.0	1/127.0/6.0	1/132.0/7.0	0/155.0/10.0	3/105.5/2.0	8/647.5/31.0
1k	1/140.5/8.0	1/113.0/4.0	1/141.0/9.0	3/113.5/3.0	3/112.0/3.0	9/620.0/27.0
5k	2/120.0/4.0	4/99.5/2.0	3/115.5/5.0	1/118.5/5.0	1/129.5/8.0	11/583.0/24.0
10k	0/144.5/9.0	**5**/104.5/3.0	1/146.5/10.0	1/124.5/6.0	1/126.0/6.0	8/646.0/34.0
25k	0/140.0/7.0	0/131.0/8.0	0/139.0/8.0	1/136.5/8.0	2/146.5/10.0	3/693.0/41.0
50k	0/148.5/10.0	2/115.0/5.0	2/119.0/6.0	4/106.0/2.0	1/131.5/9.0	9/620.0/32.0
250k	0/122.0/5.0	4/**92.0/1.0**	3/**96.5/1.0**	**6/68.0/1.0**	3/**95.5/1.0**	16/**474.0/9.0**
500k	1/108.0/3.0	2/132.0/9.0	2/113.5/4.0	1/116.0/4.0	2/120.0/5.0	8/589.5/25.0
1.5M	4/87.5/2.0	2/129.5/7.0	3/104.0/3.0	1/135.0/7.0	0/128.0/7.0	10/584.0/26.0
6.7M	**11/71.0/1.0**	1/166.5/10.0	**6**/103.0/2.0	3/137.0/9.0	**6**/115.5/4.0	**27**/593.0/26.0
RoBERTa						
0.5k	1/140.0/9.0	1/175.0/10.0	0/163.0/9.0	0/170.5/9.0	0/171.0/10.0	2/819.5/47.0
1k	3/132.5/8.0	1/160.0/9.0	2/142.0/8.0	0/165.0/8.0	2/132.0/7.0	8/731.5/40.0
5k	3/**90.0/1.0**	3/114.0/5.0	0/102.5/3.0	4/105.5/5.0	**5**/100.5/3.0	15/512.5/17.0
10k	1/125.5/6.0	1/129.0/7.0	3/112.0/6.0	1/118.5/7.0	3/118.5/6.0	9/603.5/32.0
25k	3/103.5/3.0	0/151.0/8.0	2/109.0/5.0	1/108.5/6.0	2/105.0/4.0	8/577.0/26.0
50k	**4**/99.0/2.0	1/106.5/4.0	**4/85.5/1.0**	4/83.0/2.0	1/92.5/2.0	14/**466.5/11.0**
250k	0/128.0/7.0	0/98.5/3.0	3/126.0/7.0	**8/71.0/1.0**	**5/87.0/1.0**	16/510.5/19.0
500k	2/115.0/5.0	3/88.0/2.0	1/104.0/4.0	4/102.0/4.0	1/110.5/5.0	11/519.5/20.0
1.5M	**4**/108.5/4.0	**8/72.0/1.0**	**4**/87.0/2.0	0/101.0/3.0	2/136.5/8.0	**18**/505.0/18.0
6.7M	0/168.0/10.0	4/116.0/6.0	2/179.0/10.0	0/185.0/10.0	1/156.5/9.0	7/804.5/45.0
BERTweet						
0.5k	1/142.0/7.0	0/166.5/8.0	0/169.0/9.0	0/174.0/8.0	1/153.0/8.0	2/804.5/40.0
1k	**7**/79.5/2.0	2/112.0/6.0	3/89.0/3.0	0/99.5/4.0	4/74.0/2.0	16/454.0/17.0
5k	4/**71.0/1.0**	**5**/80.0/3.0	3/**74.0/1.0**	**12/39.5/1.0**	**10/53.0/1.0**	**34/317.5/7.0**
10k	3/89.5/3.0	**5**/79.0/2.0	4/95.0/5.0	1/76.5/3.0	3/83.0/3.0	16/423.0/16.0
25k	1/94.0/4.0	1/**73.5/1.0**	3/77.0/2.0	5/71.0/2.0	0/108.5/4.0	10/424.0/13.0
50k	2/110.5/5.0	4/96.5/4.0	6/94.0/4.0	2/117.0/**6.0**	3/122.5/6.0	17/540.5/25.0
250k	2/116.5/6.0	1/106.0/5.0	0/126.5/6.0	0/116.0/5.0	0/136.0/7.0	3/601.0/29.0
500k	0/163.0/8.0	0/135.5/7.0	1/151.5/7.0	2/135.5/7.0	1/121.0/5.0	4/706.5/34.0
1.5M	0/173.0/10.0	1/170.0/9.0	0/175.0/10.0	0/178.0/9.0	0/169.0/9.0	1/865.0/47.0
6.7M	2/171.0/9.0	0/191.0/10.0	2/159.0/8.0	0/203.0/10.0	0/190.0/10.0	4/914.0/47.0

Bold values indicate the best results

Table 27 Best results achieved for each dataset by adapting the Transformer-based models with different samples of generic tweets

Dataset	Accuracy			F_1-macro		
	%	Classifier	Model	%	Classifier	Model
Iro	82.30	MLP	BERTweet-50K	75.87	LR	BERT-500K
Sar	77.32	SVM	BERT-50K	75.85	SVM	BERT-50K
Ntu	94.38	SVM	BERTweet-1.5M	94.26	SVM	BERTweet-1.5M
S15	94.18	MLP	BERTweet-1K	86.22	MLP	BERTweet-1K
Stm	93.04	SVM	BERTweet-5K	93.02	SVM	BERTweet-5K
Per	89.51	LR	BERTweet-10K	87.53	LR	BERTweet-10K
Hob	89.83	MLP	BERTweet-50K	88.30	MLP	BERTweet-50K
Iph	88.16	MLP	RoBERTa-25K	85.85	MLP	RoBERTa-25K
Mov	93.29	MLP	BERTweet-50K	88.27	LR	BERTweet-50K
San	91.83	LR	BERTweet-10K	91.77	LR	BERTweet-10K
Nar	97.04	MLP	BERTweet-1K	96.91	MLP	BERTweet-1K
Arc	92.08	LR	BERTweet-25K	91.92	LR	BERTweet-25K
S18	90.48	SVM	BERTweet-10K	90.40	SVM	BERTweet-10K
OMD	88.99	SVM	BERTweet-5K	88.17	SVM	BERTweet-5K
HCR	82.18	XGB	RoBERTa-1K	78.18	LR	BERTweet-250K
STS	95.38	MLP	BERTweet-50K	94.59	MLP	BERTweet-50K
SSt	89.60	SVM	BERTweet-10K	89.36	SVM	BERTweet-10K
Tar	87.83	SVM	BERTweet-1K	87.82	SVM	BERTweet-1K
Vad	92.80	LR	BERTweet-1K	91.64	LR	BERTweet-1K
S13	90.70	LR	BERTweet-5K	88.59	LR	BERTweet-5K
S17	93.49	SVM	BERTweet-10K	93.07	SVM	BERTweet-10K
S16	91.98	SVM	BERTweet-5K	90.30	SVM	BERTweet-5K

Table 28 Top 10 results achieved for combinations of Transformer-based models and classifiers by adapting the Transformer-based models with different samples of generic tweets

Accuracy			F_1-macro		
Model	Classifier	Avg. rank pos.	Model	Classifier	Avg. rank pos.
BERTweet-5K	LR	11.95	BERTweet-5K	LR	11.18
BERTweet-5K	MLP	14.05	BERTweet-25K	SVM	13.43
BERTweet-25K	MLP	14.64	BERTweet-10K	SVM	13.95
BERTweet-25K	LR	16.14	BERTweet-10K	LR	14.2
BERTweet-50K	MLP	16.43	BERTweet-25K	LR	14.32
BERTweet-1K	MLP	16.77	BERTweet-1K	LR	15.11
BERTweet-10K	LR	16.82	BERTweet-5K	MLP	15.95
BERTweet-25K	SVM	17.02	BERTweet-25K	MLP	16.11
BERTweet-1K	LR	17.68	BERTweet-50K	LR	16.8
BERTweet-10K	SVM	17.93	BERTweet-50K	SVM	17.43

Table 29 Top 10 results achieved by adapting the Transformer-based models with different samples of generic tweets, in terms of the average rank position

Accuracy		F_1-macro	
Model	Avg. rank pos.	Model	Avg. rank pos.
BERTweet-5K	37.76	BERTweet-5K	40.13
BERTweet-10K	40.81	BERTweet-10K	43.27
BERTweet-25K	41.24	BERTweet-25K	43.8
BERTweet-1K	43.09	BERTweet-1K	44.78
BERTweet-50K	45.22	BERTweet-50K	47.42
BERTweet-250K	48.65	BERTweet-250K	50.19
BERTweet-500K	53.96	BERTweet-500K	55.59
BERTweet-0.5K	58.0	BERTweet-0.5K	59.59
BERTweet-1.5M	66.63	BERTweet-1.5M	66.48
BERTweet-6.7M	71.09	BERTweet-6.7M	70.46

Table 30 Summary of the results for each classifier, from best to worst, by adapting the Transformer-based models with samples of generic tweets, in terms of the average rank position

Accuracy		F_1-macro	
Classifier	Avg. rank pos.	Classifier	Avg. rank pos.
MLP	48.52	LR	48.71
LR	53.17	MLP	49.92
SVM	70.74	SVM	60.83
XGB	84.9	XGB	93.16
RF	120.18	RF	124.87

formance by using smaller samples of tweets as compared to BERT and RoBERTa. Our hypothesis is that, considering that the weights in BERTweet's layers are specifically adjusted to fit tweets' language style, using more data to adapt the model means only continue the initial training. It may be that lots of data may harm the learned weights of the model. Thus, we suggest that when employ adaptive pretraining in Transformer-based models, such as BERT, RoBERTa, and BERTweet, samples of different sizes may be exploited instead of adopting a dataset with a massive number of instances.

Additionally, we present a comparison among all adapted Transformer-based models against their original versions. Tables 31, 32, and 33 report this comparison in terms of the average rank position for BERT, RoBERTa, and BERTweet, respectively. We can see that the adapted versions achieved meaningful predictive performances as compared to their original models, which indicates that adaptive pretraining strategies can boost classification performance in Twitter sentiment analysis. Moreover, from Tables 31 and 32, we note that the adapted versions of BERT and RoBERTa benefited most from samples containing a large amount of tweets. Conversely, as pointed out before, BERTweet achieved better overall performances by using smaller samples, as shown in Table 33.

Table 31 Comparison among all adapted BERT models and BERT's original version (no adaptation), in terms of the average rank position

Accuracy		F_1-macro	
Model	Avg. rank pos.	Model	Avg. rank pos.
BERT-250K	25.62	BERT-250K	25.62
BERT-5K	26.95	BERT-1.5M	26.45
BERT-1.5M	26.96	BERT-6.7M	26.69
BERT-500K	27.09	BERT-500K	26.7
BERT-6.7M	27.67	BERT-5K	27.69
BERT-0.5K	28.16	BERT-50K	28.36
BERT-50K	28.38	BERT-0.5K	28.4
BERT-1K	28.46	BERT-1K	28.95
BERT-10K	29.52	BERT (original)	29.5
BERT (original)	29.52	BERT-10K	29.68
BERT-25K	29.66	BERT-25K	29.95

Table 32 Comparison among all adapted RoBERTa models and RoBERTa's original version (no adaptation), in terms of the average rank position

Accuracy		F_1-macro	
Model	Avg. rank pos.	Model	Avg. rank pos.
RoBERTa-50K	24.34	RoBERTa-50K	24.24
RoBERTa-500K	24.69	RoBERTa-1.5M	24.61
RoBERTa-1.5M	24.82	RoBERTa-500K	24.95
RoBERTa-5K	25.44	RoBERTa-5K	25.54
RoBERTa-250K	25.53	RoBERTa-250K	25.66
RoBERTa-25K	26.49	RoBERTa-25K	27.05
RoBERTa-10K	27.28	RoBERTa-10K	27.50
RoBERTa-1K	29.84	RoBERTa-1K	29.65
RoBERTa-0.5K	32.01	RoBERTa-0.5K	31.78
RoBERTa-6.7M	32.75	RoBERTa-6.7M	31.96
RoBERTa (original)	34.81	RoBERTa (original)	35.06

Addressing research question RQ3, we could see that adaptive pretraining of Transformer-based models improves the classification effectiveness in Twitter sentiment analysis. Nevertheless, using large sets of tweets does not guarantee better predictive performances, particularly for those models trained from scratch on tweets, such as BERTweet. We could observe that BERTweet benefited most from samples of tweets containing 50K tweets or less. Furthermore, regarding the classifiers, in general, MLP and LR seem to be good choices of classifiers to be employed after extracting the features from adapted Transformer-based models.

Table 33 Comparison among all adapted BERTweet models and BERTweet's original version (no adaptation), in terms of the average rank position

Accuracy		F_1-macro	
Model	Avg. rank pos.	Model	Avg. rank pos.
BERTweet-5K	20.74	BERTweet-5K	21.38
BERTweet-25K	22.62	BERTweet-25K	22.94
BERTweet-10K	22.85	BERTweet-10K	23.10
BERTweet-1K	23.73	BERTweet-1K	23.93
BERTweet-50K	25.25	BERTweet-50K	25.55
BERTweet-250K	26.67	BERTweet-250K	26.66
BERTweet-500K	30.31	BERTweet-500K	30.48
BERTweet-0.5K	31.70	BERTweet-0.5K	31.72
BERTweet (original)	33.80	BERTweet (original)	33.35
BERTweet-1.5M	34.80	BERTweet-1.5M	34.05
BERTweet-6.7M	35.53	BERTweet-6.7M	34.85

7 Adapting transformer-based models to sentiment datasets

The experiments conducted in this section aim at answering the research question RQ4, stated as follows:

RQ4 Can Transformer-based autoencoder models benefit from a second phase of adaptive pretraining procedure with tweets from sentiment analysis datasets?

We address this research question by evaluating whether the sentiment classification of tweets benefits from adapting language models to tweets from sentiment analysis datasets. For this purpose, we use the same collection of 22 benchmark datasets presented in Sect. 3.1 (Table 1). We perform this evaluation by assessing three distinct strategies to simulate three real-world scenarios. In addition, as done in Sect. 6, all experiments were performed three times using different seeds (12, 34, 56), with all the same hyperparameter and we report the average of the results.

The first adaptation strategy we investigate, referred to as *InData*, simulates the usage of a specific sentiment dataset itself as the new domain dataset to adapt a pretrained language model. Precisely, each one of the 22 datasets is used once as the target dataset. For each of the 22 datasets, we use a 10-fold cross-validation procedure. In each of the ten executions, we use the tweets from nine folds as the source data (i.e., the training data) used to adjust a language model, which is then validated on the one remaining part of the data, referred to as the target dataset (i.e., the test data).

The second strategy, referred to as *LOO* (Leave One dataset Out), aims at simulating the situation where a collection of general sentiment datasets is available to adapt the language model. We use each dataset once as the target dataset while the tweets from the remaining 21 datasets are combined to adjust the language model. Although the target dataset contains sentiment labels for each tweet, these labels are not used in the adaptation process as we leverage the intermediate self-supervised masked language model task to tune the network parameters.

The third and last strategy, referred to as *AllData*, is a combination of the two others. Specifically, as for strategy InData, for each assessed dataset (target dataset), and for each of the nine folds in the 10-fold cross-validation procedure, we combine the tweets from the nine folds (i.e., the training data of the target dataset) with the tweets from the remaining 21 datasets to adapt the language model. This last strategy evaluates the benefits of combining the tweets from a specific sentiment target dataset with a representative general sentiment dataset corpus in the adaptation process.

Tables 34, 35, and 36 present the predictive performances achieved by adapting BERT, RoBERTa, and BERTweet, respectively, with strategies InData, LOO, and AllData, one at a time, by using the SVM classifier. As in previous sections, for space constraints, we only report the detailed evaluation using the SVM classifier.

From Table 34, we can observe that, although BERT seems to benefit most from strategy InData, which uses only the target dataset itself to adjust the language models, the Friedman and the Nemenyi tests did not detect any significant differences between strategies InData, LOO, and AllData. Regarding RoBERTa and BERTweet models (Tables 35 and 36, respectively), adapting them using strategies that combine tweets from distinct sentiment analysis corpora achieved the best results for most datasets. More clearly, AllData, which combines the tweets from the target dataset and tweets from a collection of sentiment datasets, achieved the best overall results with both RoBERTa and BERTweet. As a matter of fact, the Friedman and the Nemenyi tests indicate that strategy AllData with RoBERTa outperformed strategy InData with statistical difference between them. Similarly, strategies AllData and LOO with BERTweet are significantly better than strategy InData. It is also noteworthy that smaller datasets seem to have benefited most from adapting RoBERTa and BERTweet by using strategy LOO. On the other hand, larger datasets achieved higher predictive performances when using strategy AllData to fine-tune RoBERTa and BERTweet. Tables 37 and 38 show a summary of the complete evaluation regarding all classifiers in terms of classification accuracy and F_1-macro, respectively.

Regarding the overall results achieved for each dataset, Table 39 presents the best results. We can note that when adapting the Transformer-based models with tweets from sentiment datasets, BERTweet outperformed BERT and RoBERTa for all datasets, except for datasets sarcasm (sar) and hobbit (hob). Interestingly, as mentioned before, while strategy LOO achieved the best results for smaller datasets, larger datasets seem to benefit from strategy AllData. Precisely, strategy AllData achieved the best overall performances in ten out of the 22 datasets in terms of accuracy and in 11 out of the 22 datasets in terms of F_1-macro. Strategy LOO achieved the best results in nine out of the 22 datasets for both accuracy and F_1-macro. The better performance of the AllData strategy for larger target datasets indicates that the significant amount of information present in the target dataset is indispensable for the adaptation process, while the information present in smaller datasets seems not to contribute to the adaptation process, making the LOO strategy adequate for datasets with a limited amount of tweets.

Conversely, strategy InData did not achieve meaningful results. The inferior performance of the InData strategy in almost all datasets shows that, regardless of the size of the dataset, the use of external and more extensive data brings more information to the adaptation process, improving the final performance.

Table 34 Accuracies and F_1-macro scores (%) achieved by evaluating BERT with adaptation strategies InData, LOO, and AllData using the SVM classifier

Dataset	Accuracy			F_1-macro		
	AllData	LOO	InData	AllData	LOO	InData
Iro	74.40	**78.81**	67.90	65.40	**70.55**	59.60
Sar	**71.0**	70.18	64.10	68.50	**68.58**	60.20
Ntu	85.00	82.74	**88.10**	84.70	82.39	**87.80**
S15	89.70	88.14	**89.80**	77.50	77.11	**77.80**
Stm	88.80	**90.25**	89.90	88.70	**90.24**	89.80
Per	84.40	**85.66**	82.00	82.20	**83.49**	80.00
Hob	**84.60**	84.46	82.30	82.90	**83.08**	80.70
Iph	82.70	**83.09**	83.00	80.80	81.07	**81.40**
Mov	85.80	**86.46**	84.60	79.40	**80.14**	78.10
San	87.60	87.49	**87.80**	87.50	87.43	**87.70**
Nar	92.20	92.50	**94.90**	92.00	92.25	**94.70**
Arc	89.10	88.42	**89.90**	88.90	88.20	**89.80**
S18	87.70	87.36	**89.70**	87.60	87.26	**89.60**
OMD	85.90	85.73	**87.30**	85.00	84.74	**86.40**
HCR	79.30	79.03	**79.60**	75.70	75.25	**75.90**
STS	91.70	90.71	**93.50**	90.50	89.35	**92.60**
SSt	84.70	84.71	**87.50**	84.30	84.39	**87.20**
Tar	85.70	86.24	**86.90**	85.70	86.23	**86.90**
Vad	89.90	90.16	**91.50**	88.50	88.84	**90.30**
S13	87.50	87.60	**88.70**	85.20	85.31	**86.60**
S17	91.80	91.56	**92.90**	91.30	91.04	**92.40**
S16	89.50	89.07	**90.70**	87.40	86.93	**88.80**
#wins	2	5	**15**	0	6	**16**
Rank sums	49.0	49.0	**34.0**	51.0	48.0	**33.0**
Position	2.5	2.5	**1.0**	3.0	2.0	**1.0**

Bold values indicate the best results

Next, we present an overall evaluation of combining all adapted models and classifiers across the 22 datasets, in terms of the average rank position. Table 40 reports the top ten results among all 45 possible combinations (3 language models $\times$ 3 adaptation strategies $\times$ 5 classification algorithms). We can observe that the LR classifier trained with BERTweet embeddings adapted via strategy AllData achieved the best overall predictive performances. Also, note that the fine-tuned BERTweet embeddings with strategies AllData and LOO, combined with LR, MLP, and SVM, appear at the top of the ranking (top six results). Another point worth highlighting is that BERTweet masters the top ten results, appearing in eight out of the ten positions in terms of accuracy and in nine out of the ten positions in terms of F_1-macro.

Tables 41 and 42 show the results among all adapted models and a summary of the results for each classifier, from best to worst, respectively, in terms of the average rank

Table 35 Accuracies and F_1-macro scores (%) achieved by evaluating RoBERTa with adaptation strategies InData, LOO, and AllData using the SVM classifier

Dataset	Accuracy			F_1-macro		
	AllData	LOO	InData	AllData	LOO	InData
Iro	46.70	46.67	46.70	31.00	31.00	31.00
Sar	64.00	**65.00**	64.00	52.90	53.49	**54.20**
Ntu	**84.30**	83.48	81.20	**83.90**	83.07	80.80
S15	**87.20**	86.31	86.20	**74.90**	74.28	72.40
Stm	90.00	**90.25**	87.60	89.90	**90.21**	87.60
Per	**71.10**	70.38	65.50	**70.20**	69.17	64.90
Hob	72.80	**73.94**	71.30	71.90	**73.10**	70.50
Iph	**79.90**	78.96	78.60	**78.60**	77.72	77.20
Mov	**81.50**	79.87	72.10	**75.30**	74.01	66.50
San	**87.50**	87.17	85.50	**87.40**	87.04	85.30
Nar	**93.10**	92.50	92.10	**92.90**	92.35	91.90
Arc	89.40	**89.47**	89.00	**89.30**	89.27	88.70
S18	88.40	**88.54**	88.00	88.30	**88.44**	87.80
OMD	85.60	84.58	**85.70**	84.50	83.60	**84.70**
HCR	76.90	76.04	**78.10**	73.20	72.59	**74.20**
STS	**92.60**	92.13	91.50	**91.60**	90.97	90.30
SSt	**86.40**	85.63	85.90	**86.10**	85.41	85.70
Tar	85.90	85.87	**86.30**	85.90	85.85	**86.30**
Vad	89.80	89.37	**89.90**	88.60	88.10	88.60
S13	**86.60**	86.41	86.10	**84.60**	84.41	83.90
S17	**92.00**	91.71	91.60	**91.50**	91.21	91.10
S16	89.50	**89.71**	89.30	87.60	**87.75**	87.40
#wins	**12**	6	5	**13**	4	5
Rank sums	**33.0**	44.0	55.0	**32.5**	45.0	54.5
Position	**1.0**	2.0	3.0	**1.0**	2.0	3.0

{AllData} $\succ$ {InData}
Bold values indicate the best results

position. Once again, from Table 41, we can notice that all BERTweet adapted models (InData, LOO, and AllData) were ranked in the top three results. Among the classifiers, as we can see in Table 42, MLP and LR achieved the best predictive performances and were ranked as the top two best classifiers. Conversely, RF was ranked as the worst classifier.

To evaluate the effectiveness of adapting the Transformer-based models using tweets from sentiment datasets, we present a comparison among all the adaptation strategies assessed in this study for each language model. Specifically, we compare the adapted models presented in this section, by using strategies InData, LOO, and AllData, against the best adapted models identified in Sect. 6, i.e., BERT-250K,

Table 36 Accuracies and F_1-macro scores (%) achieved by evaluating BERTweet with adaptation strategies InData, LOO, and AllData using the SVM classifier

Dataset	Accuracy			F_1-macro		
	AllData	LOO	InData	AllData	LOO	InData
Iro	74.60	**83.10**	66.70	66.50	**77.24**	60.10
Sar	**68.60**	67.50	61.80	**65.50**	64.32	56.40
Ntu	92.10	**93.54**	90.10	91.80	**93.33**	89.80
S15	90.70	**92.84**	90.20	80.00	**84.76**	78.60
Stm	92.70	**92.75**	90.50	92.60	**92.73**	90.50
Per	86.10	**86.55**	82.50	84.10	**84.48**	80.70
Hob	87.10	**87.17**	82.50	85.60	**85.62**	80.80
Iph	**85.10**	83.48	83.30	**83.50**	81.79	81.80
Mov	**89.90**	88.42	87.00	**84.50**	81.99	80.90
San	**91.40**	91.34	89.00	**91.40**	91.27	88.90
Nar	**97.00**	96.66	96.20	**96.80**	96.54	96.00
Arc	**91.40**	90.57	90.70	**91.30**	90.40	90.50
S18	**90.90**	90.26	90.60	**90.80**	90.19	90.50
OMD	89.20	**89.77**	88.40	88.40	**88.99**	87.50
HCR	**81.50**	81.27	80.40	**77.90**	77.79	76.70
STS	**95.20**	94.99	94.70	**94.50**	94.21	93.90
SSt	**89.10**	88.51	88.80	**88.90**	88.25	88.50
Tar	**87.70**	87.63	87.30	**87.70**	87.62	87.30
Vad	92.50	**92.73**	92.30	91.40	**91.70**	91.20
S13	**90.00**	89.52	89.40	**88.00**	87.49	87.40
S17	**93.60**	93.59	93.20	93.10	**93.17**	92.80
S16	**91.80**	91.62	91.50	**90.10**	89.90	89.80
#wins	**14**	8	0	**13**	9	0
Rank sums	**30.0**	39.0	63.0	**31.0**	39.0	62.0
Position	**1.0**	2.0	3.0	**1.0**	2.0	3.0

{AllData, LOO} $\succ$ {InData}
Bold values indicate the best results

RoBERTa-50K, and BERTweet-5K. Table 43 reports these results in terms of the average rank position for BERT, RoBERTa, and BERTweet.

Regarding BERT, as shown in Table 43, note that all the adaptation strategies using tweets from sentiment datasets achieved better overall results than using the sample of 250K generic tweets. Moreover, strategy InData appears at the top of the ranking as the best adaptation strategy. It is worth mentioning that strategy InData uses only the tweets from the target dataset itself to adjust the language model. This means that the strategy InData used a number of tweets much smaller than the 250K tweets contained in the sample. On the other hand, strategy InData did not achieve meaningful results for RoBERTa and BERTweet models. Nevertheless, for these models, strategies AllData

Table 37 Overview of the results (number of wins, rank sum, and rank position, respectively) achieved by each classifier when adapting the Transformer-Autoencoder models using strategies InData, LOO, and AllData in terms of accuracy

Strategy	LR	SVM	MLP	RF	XGB	Total
BERT						
AllData	1/51.0/2.0	2/49.0/2.5	3/43.5/2.0	5/42.0/2.0	2/47.5/2.0	13/233.0/10.5
LOO	1/56.0/3.0	5/49.0/2.5	3/56.5/3.0	6/49.0/3.0	3/53.0/3.0	18/263.5/14.5
InData	**20/25.0/1.0**	**15/34.0/1.0**	**14/32.0/1.0**	**9/41.0/1.0**	**14/31.5/1.0**	**72/163.5/5.0**
RoBERTa						
AllData	**11/32.0/1.0**	**11/33.0/1.0**	**10/35.0/1.0**	**11/34.0/1.0**	**12/35.0/1.0**	**55/169.0/5.0**
LOO	6/48.0/2.0	6/44.0/2.0	9/42.0/2.0	10/37.0/2.0	8/41.0/2.0	39/212.0/10.0
InData	3/52.0/3.0	4/55.0/3.0	2/55.0/3.0	1/61.0/3.0	2/56.0/3.0	12/279.0/15.0
BERTweet						
AllData	**10/36.5/1.0**	**14/30.0/1.0**	9/**34.5/1.0**	**13/31.0/1.0**	**12/33.5/1.0**	**58/165.5/5.0**
LOO	9/39.5/2.0	8/39.0/2.0	**11**/37.0/2.0	8/39.5/2.0	8/39.0/2.0	44/194.0/10.0
InData	2/56.0/3.0	0/63.0/3.0	1/60.5/3.0	1/61.5/3.0	1/59.5/3.0	5/300.5/15.0

Bold values indicate the best results

Table 38 Overview of the results (number of wins, rank sum, and rank position, respectively) achieved by each classifier when adapting the Transformer-Autoencoder models using strategies InData, LOO, and AllData in terms of F_1-macro

Strategy	LR	SVM	MLP	RF	XGB	Total
BERT						
AllData	1/51.0/2.0	0/51.0/3.0	3/45.0/2.0	6/41.0/2.0	3/48.5/2.0	13/236.5/11.0
LOO	2/56.0/3.0	6/48.0/2.0	4/55.0/3.0	5/51.0/3.0	3/52.0/3.0	20/262.0/14.0
InData	**19/25.0/1.0**	**16/33.0/1.0**	**15/32.0/1.0**	**11/40.0/1.0**	**15/31.5/1.0**	**76/161.5/5.0**
RoBERTa						
AllData	**13/31.0/1.0**	**12/32.5/1.0**	9/**35.5/1.0**	**11/34.0/1.0**	**12/35.0/1.0**	**57/168.0/5.0**
LOO	5/49.0/2.0	4/45.0/2.0	**10**/42.0/2.0	10/36.0/2.0	8/40.0/2.0	37/212.0/10.0
InData	4/52.0/3.0	4/54.5/3.0	2/54.5/3.0	1/62.0/3.0	2/57.0/3.0	13/280.0/15.0
BERTweet						
AllData	**10/35.5/1.0**	**13/31.0/1.0**	**13/31.0/1.0**	**13/31.0/1.0**	**10/35.0/1.0**	**59/163.5/5.0**
LOO	**10/37.5/2.0**	9/39.0/2.0	9/39.0/2.0	8/39.5/2.0	**10**/37.5/2.0	46/192.5/10.0
InData	1/59.0/3.0	0/62.0/3.0	0/62.0/3.0	1/61.5/3.0	0/59.5/3.0	2/304.0/15.0

Bold values indicate the best results

Table 39 Best results achieved for each dataset by adapting the Transformer-based models using strategies InData, LOO, and AllData

Dataset	Accuracy			F_1-macro		
	%	Classifier	Model	%	Classifier	Model
Iro	83.33	LR	BERTweet-LOO	77.24	SVM	BERTweet-LOO
Sar	80.66	MLP	RoBERTa-LOO	79.35	MLP	RoBERTa-LOO
Ntu	93.89	LR	BERTweet-LOO	93.66	LR	BERTweet-LOO
S15	94.39	MLP	BERTweet-LOO	87.31	MLP	BERTweet-LOO
Stm	92.75	SVM	BERTweet-LOO	92.73	SVM	BERTweet-LOO
Per	88.52	MLP	BERTweet-LOO	86.14	LR	BERTweet-LOO
Hob	89.5	MLP	RoBERTa-InData	88.0	MLP	RoBERTa-InData
Iph	87.8	LR	BERTweet-InData	85.8	LR	BERTweet-AllData
Mov	91.1	LR	BERTweet-AllData	85.4	LR	BERTweet-AllData
San	91.6	LR	BERTweet-AllData	91.5	LR	BERTweet-AllData
Nar	97.0	SVM	BERTweet-AllData	96.8	MLP	BERTweet-AllData
Arc	92.1	MLP	BERTweet-AllData	91.9	MLP	BERTweet-AllData
S18	90.9	SVM	BERTweet-AllData	90.8	SVM	BERTweet-AllData
OMD	89.77	SVM	BERTweet-LOO	88.99	SVM	BERTweet-LOO
HCR	82.21	XGB	BERTweet-LOO	77.9	SVM	BERTweet-AllData
STS	95.2	SVM	BERTweet-AllData	94.5	SVM	BERTweet-AllData
SSt	89.1	SVM	BERTweet-AllData	88.9	SVM	BERTweet-AllData
Tar	87.7	SVM	BERTweet-AllData	87.7	SVM	BERTweet-AllData
Vad	93.14	LR	BERTweet-LOO	92.05	LR	BERTweet-LOO
S13	90.4	LR	BERTweet-InData	88.3	LR	BERTweet-InData
S17	93.6	SVM	BERTweet-AllData	93.17	SVM	BERTweet-LOO
S16	91.8	SVM	BERTweet-AllData	90.1	SVM	BERTweet-AllData

Table 40 Top 10 results achieved for combinations of Transformer-based models and classifiers by adapting the models using strategies InData, LOO, and AllData

Accuracy			F_1-macro		
Model	Classifier	Avg. rank pos.	Model	Classifier	Avg. rank pos.
BERTweet-AllData	LR	4.86	BERTweet-AllData	LR	4.18
BERTweet-AllData	MLP	5.57	BERTweet-LOO	LR	5.34
BERTweet-LOO	MLP	6.07	BERTweet-AllData	MLP	5.45
BERTweet-LOO	LR	6.16	BERTweet-AllData	SVM	6.11
BERTweet-AllData	SVM	7.36	BERTweet-LOO	MLP	6.75
BERTweet-LOO	SVM	8.64	BERTweet-LOO	SVM	6.86
BERTweet-InData	LR	9.11	BERTweet-InData	LR	8.36
BERTweet-InData	MLP	11.27	RoBERTa-AllData	LR	11.89
RoBERTa-AllData	MLP	13.09	BERTweet-InData	MLP	11.95
RoBERTa-AllData	LR	13.48	BERTweet-InData	SVM	12.93

Table 41 Comparison among all adapted Transformer-based models using strategies InData, LOO, and AllData, in terms of the average rank position

Accuracy		F_1-macro	
Model	Avg. rank pos.	Model	Avg. rank pos.
BERTweet-AllData	12.99	BERTweet-AllData	13.51
BERTweet-LOO	14.04	BERTweet-LOO	14.46
BERTweet-InData	19.59	BERTweet-InData	19.95
RoBERTa-AllData	22.76	RoBERTa-AllData	22.70
RoBERTa-LOO	24.46	RoBERTa-LOO	24.00
BERT-InData	24.53	BERT-InData	24.32
RoBERTa-InData	27.90	RoBERTa-InData	27.70
BERT-AllData	30.12	BERT-AllData	29.85
BERT-LOO	30.61	BERT-LOO	30.51

Table 42 Summary of the results for each classifier, from best to worst, by adapting the Transformer-based models using strategies InData, LOO, and AllData, in terms of the average rank position

Accuracy		F_1-macro	
Classifier	Avg. rank pos.	Classifier	Avg. rank pos.
MLP	14.83	LR	14.41
LR	15.73	MLP	15.28
SVM	22.42	SVM	19.77
XGB	25.70	XGB	28.13
RF	36.32	RF	37.41

and LOO, which also use tweets from sentiment datasets, achieved rather comparable performances and were ranked as the top two best adaptation strategies.

To acknowledge the effectiveness of adapting the Transformer-based models to tweets from sentiment datasets, i.e., using the strategies InData, LOO, and AllData, we present an overall comparison among these strategies and the 47 models assessed in this study (Sects. 4, 5, and 6). Tables 44 and 45 present, respectively, the ten best and the ten worst combinations of models and classifiers, in terms of the average rank position, regarding all 280 combinations of models and classifiers (56 models and five classifiers). We note that BERTweet tuned with tweets from sentiment datasets and combined with LR and MLP had the four best results, in terms of accuracy, and the two best results, in terms of F_1-macro. These combinations were followed by BERTweet tuned with generic tweets. More specifically, combinations with the strategy AllData and LOO achieved better overall results. Independently of the language model, LR and MLP were the most frequent classifier in the top 10 results. Conversely, all the ten worst combinations are static representations combined with RF, which was unanimous in the worst model and classifier combinations.

Disregarding the classifiers, Tables 46 and 47 present the top ten and the bottom ten models, respectively, by comparing all 56 word representations assessed in this study (14 static representations + 3 Transformer-based models + 30 models adapted with

Table 43 Comparison among the adapted models by using strategies InData, LOO, and AllData, against the best adapted models with different samples of generic tweets

Accuracy		F_1-macro	
Model	Avg. rank pos.	Model	Avg. rank pos.
BERT			
BERT-InData	8.04	BERT-InData	8.10
BERT-AllData	10.61	BERT-AllData	10.66
BERT-LOO	11.24	BERT-LOO	11.29
BERT-250K	12.12	BERT-250K	11.95
RoBERTa			
RoBERTa-AllData	8.81	RoBERTa-AllData	8.85
RoBERTa-LOO	9.89	RoBERTa-LOO	9.84
RoBERTa-50K	11.61	RoBERTa-50K	11.52
RoBERTa-InData	11.68	RoBERTa-InData	11.80
BERTweet			
BERTweet-AllData	9.00	BERTweet-AllData	9.11
BERTweet-LOO	9.76	BERTweet-LOO	9.76
BERTweet-5K	10.19	BERTweet-5K	10.20
BERTweet-InData	13.05	BERTweet-InData	12.93

samples of generic tweets + 9 models tuned with sentiment datasets). From Table 46, we can acknowledge the good performance of adapting the Transformer-based models using tweets from sentiment datasets. Specifically, the adapted BERTweet models using strategies AllData and LOO appear at the top of the ranking as the two best models. We can also notice that adapting BERTweet with generic tweets results in performance improvement to BERTweet. Regarding the bottom ten models, from Table 47, we can see that all of them are static representations.

Lastly, regarding research question RQ4, we can highlight that adapting Transformer-based models using tweets from sentiment datasets seems to boost classification performance in Twitter sentiment analysis. As a matter of fact, the strategies AllData and LOO exploited in this section, which use a collection of sentiment tweets to adjust a language model, achieved better overall results than using samples of unlabeled, or generic unlabeled, tweets. Although we do not use the labels of those tweets in the adaptation procedure, they may carry a lot of sentiment information as compared to the tweets from the Edinburgh corpus, which originated the samples of generic unlabeled tweets used in the experiments. Furthermore, BERTweet embeddings adapted with the AllData strategy seems to be very effective in determining the sentiment expressed in tweets, especially when used to train LR, MLP, and SVM classifiers.

Table 44 Top 10 results achieved by evaluating combinations of models and classifiers, regarding all 56 models assessed in this study

Accuracy			F_1-macro		
Model	Classifier	Avg. rank pos.	Model	Classifier	Avg. rank pos.
BERTweet-LOO	LR	17.30	BERTweet-LOO	LR	15.52
BERTweet-AllData	LR	18.16	BERTweet-AllData	LR	16.30
BERTweet-LOO	MLP	18.55	BERTweet-5K	LR	18.64
BERTweet-AllData	MLP	19.36	BERTweet-AllData	MLP	20.68
BERTweet-5K	LR	20.64	BERTweet-LOO	MLP	21.00
BERTweet-5K	MLP	23.14	BERTweet-25K	SVM	22.48
BERTweet-25K	MLP	24.30	BERTweet-10K	SVM	23.16
BERTweet-25K	LR	26.23	BERTweet-25K	LR	23.30
BERTweet-1K	MLP	26.91	BERTweet-10K	LR	23.86
BERTweet-50K	MLP	27.02	BERTweet-AllData	SVM	24.82

Table 45 Bottom 10 results achieved by evaluating combinations of models and classifiers, regarding all 56 models assessed in this study

Accuracy			F_1-macro		
Model	Classifier	Avg. rank pos.	Model	Classifier	Avg. rank pos.
EWE	RF	247.23	DeepMoji	RF	252.86
W2V-Araque	RF	249.75	BERTweet-static	LR	253.14
W2V-GN	RF	250.00	EWE	RF	256.48
GloVe-WP	RF	253.68	W2V-Araque	RF	259.80
fastText	RF	255.75	W2V-GN	RF	261.70
BERT-static	RF	257.32	GloVe-WP	RF	263.11
RoBERTa-static	RF	259.70	fastText	RF	266.95
BERT-static	LR	263.34	BERT-static	RF	267.75
BERTweet-static	RF	265.91	RoBERTa-static	RF	269.93
BERTweet-static	LR	274.43	BERTweet-static	RF	275.43

Table 46 Top 10 models among the 56 word representation models assessed in this study, in terms of the average rank position

Accuracy		F_1-macro	
Model	Avg. rank pos.	Model	Avg. rank pos.
BERTweet-AllData	53.83	BERTweet-AllData	60.67
BERTweet-LOO	56.52	BERTweet-LOO	62.59
BERTweet-5K	60.56	BERTweet-5K	66.21
BERTweet-25K	65.38	BERTweet-25K	71.07
BERTweet-10K	65.51	BERTweet-10K	71.17
BERTweet-1K	68.59	BERTweet-1K	73.32
BERTweet-50K	72.31	BERTweet-50K	77.50
BERTweet-250K	78.13	BERTweet-250K	82.90
BERTweet-InData	83.80	BERTweet-InData	87.93
BERTweet-500K	86.10	BERTweet-500K	90.85

Table 47 Bottom 10 models among the 56 word representation models assessed in this study, in terms of the average rank position

Accuracy		F_1-macro	
Model	Avg. rank pos.	Model	Avg. rank pos.
SSWE	209.69	w2v-GN	204.38
GloVe-TWT	215.62	GloVe-TWT	207.22
DeepMoji	217.13	DeepMoji	208.08
EWE	217.46	EWE	208.43
TF-IDF	220.83	GloVe-WP	215.45
BERT-static	224.61	FastText	218.05
GloVe-WP	225.94	BERT-static	218.40
FastText	227.80	w2v-Araque	222.85
w2v-Araque	230.56	TF-IDF	224.34
BERTweet-static	244.21	BERTweet-static	237.01

8 Conclusions and future works

In this article, we presented an extensive assessment of modern and classical word representations when used for the task of Twitter sentiment analysis. Specifically, we assessed the classification performance of 14 static representations, the most recent Transformer-based autoencoder models, including BERT, RoBERTa, and BERTweet, as well as different adaptation strategies of the language representation tasks in such models. All models were evaluated in the context of Twitter sentiment analysis using a rich set of 22 datasets and five classifiers from distinct natures. The main focus of this study was on identifying the most appropriate word representations for the sentiment analysis of English tweets.

Based on the results of the experiments performed in this study, we can highlight the following conclusions and recommendations:

- Considering a limited computing resource scenario where static representations could play an important role, we noticed that Emo2Vec, w2v-Edin, and RoBERTa models seem to be well-suited static representations for determining the sentiment expressed in tweets. Although there is no significant difference between them, they are significantly better than many of the other assessed static representations. The good performance achieved by Emo2Vec and w2v-Edin indicates that being trained from scratch with tweets can boost the classification performance of static representations when applied in Twitter sentiment analysis. Although RoBERTa was not trained from stratch with tweets, it is a Transformer-based autoencoder model, which holds state-of-the-art performance in several NLP tasks. Regarding the classifiers, we could see that SVM and MLP achieved the best overall performances, especially when used to train RoBERTa's static embeddings. Nevertheless, in such scenario, we acknowledge that there is no global optimum language model. In that case, when implementing a classification system, we recommend the user to perform an assessment of RoBERTa, Emo2Vec, and w2v-Edin language models. Moreover, we suggest analyzing combinations of those language models with SVM and MLP classifiers.
- Regarding the Transformer-based models, we observed that BERTweet is the most appropriate language model to be used in the sentiment classification of tweets, achieving significantly better results than RoBERTa and BERT. Specifically, the particular vocabulary tweets contain, combined with a language model that was trained focused on learning their intrinsic structure, can effectively improve the performance of the Twitter sentiment analysis task. Considering the combination of language models and classifiers, we can point out that BERTweet achieved the best overall results when combined with LR and MLP. Furthermore, by comparing the Transformer-based models and the static representations, we could notice that the adaptation of the tokens' embeddings to the context they appear performed by the Transformer-based models benefits the sentiment classification task. In this context, considering a scenario where the availability of computing resources would not be an issue, we recommend BERTweet as the language model to be adopted in a Twitter sentiment classification system, being LR and MLP reasonable choices of classifiers.
- When adapting the Transformer-based pre-trained models to a large set of English unlabeled tweets, we noticed that although it improves the classification performance, using as many tweets as possible does not necessarily mean better results. Based on that, we presented an extensive evaluation of sets of tweets with different sizes, varying from 0.5K to 1.5M. These results have shown that while BERT and RoBERTa achieved better predictive performances when tuned with sets of 250K and 50K tweets, respectively, BERTweet outperformed all adapted models using only 5K tweets. Although the Friedman and the Nemenyi tests did not detect any significant difference among these results, we believe that models trained from scratch with tweets, such as BERTweet, need fewer tweets to improve their performance. Moreover, by comparing all adapted models taking into account

the classifiers, BERTweet combined with MLP, LR, and SVM achieved the best overall performances. In this context, if adapting a language model is an option, having enough computing resources at hand as well as a considerable amount of English unlabeled tweets, we recommend that the user evaluate the performance of a Twitter sentiment classification system by trying sets of tweets with different sizes. Besides, we suggest the usage of BERTweet as the language model.

– Analyzing the adaption of the language model based on Transformers autoencoders with sentiment analysis datasets, i.e., with tweets that express polarity, we can see that the adapted models' performance is better than when adapted with generic tweets. All adaption strategies with sentiment analysis datasets performed better than the best-tuned models adjusted with generic tweets. We conclude then that it is worth adapting a model based on Transformer autoencoders using a set of sentiment tweets. Among the adaption strategies – using sentiment analysis tweets – explored in the study, it was possible to perceive that each Transformer model presented a better performance with different adjustment methods. The use of only the target dataset, for example, was a good option to be used with BERT. For RoBERTa and BERTweet, the combination of the target dataset with a set of tweets from other datasets presented a good strategy for adapting the language model. In a general comparison, we noticed that BERTweet tuned with the union of the target dataset and the set of sentiment analysis tweets (BERTweet-AllData) performed better than the other adjusted models. Besides, we could observe that BERTweet-AllData presented a good performance when combined with LR and MLP classifiers. Hence, considering a scenario where a specific dataset of English tweets carrying positive and negative polarities is available for adapting a language model, we recommend using BERTweet adapted with strategy AllData as the language model of a sentiment classification system.
– After answering our research questions, we can briefly state that: (i) Transformer-based autoencoder models perform better than static representation, (ii) Transformer autoencoder models adapted to English tweets behavior better than the respective original models and, finally, (iii) it is worth adapting a language model originally trained with generic English tweets with tweets from sentiment analysis datasets. Considering all original and adapted models, the best overall performance for the English tweets sentiment analysis task was achieved by the Transformer-Autoencoder model trained from scratch with generic tweets (BERTweet) when adapted with tweets from a target sentiment dataset added by tweets from a large set of other sentiment datasets. This strategy was called BERTweet-AllData, which we consider a good suggestion for sentiment classification of English tweets, mainly when combined with MLP or LR classifiers.

For future work, we plan to investigate other methods for adjusting the language models, mainly fine-tuning them to the polarity classification as the downstream tuning task. Transformer-Autoencoder pre-trained models, like BERT, RoBERTa and BERTweet, can have its weights adjusted looking for becoming more accurate in a specific task, like sentiment analysis. This adjustment is made by adding an extra classification layer in the top of the model and back-propagating the error in the final task

through language models' weights. We intend then to compare the best results obtained in this study with the ones achieved by this specific-task category of fine-tuning.

Acknowledgements The authors would like to thank the Brazilian Research agencies FAPERJ and CNPq for the financial support.

Funding The funding was provided by Conselho Nacional de Desenvolvimento Científico e Tecnológico (Grant Nos. 310444/2018-7, 311275/2020-6, 421608/2018-8), Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro (Grant No. E26/202.914/2019 (247109)).

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

Adhikari A, Ram A, Tang R, Lin J (2019) Docbert: BERT for document classification. arxiv: 1904.08398

Agrawal A, An A, Papagelis M (2018) Learning emotion-enriched word representations. In: Proceedings of the 27th International Conference on Computational Linguistics, Association for Computational Linguistics, Santa Fe, New Mexico, USA, pp 950–961, https://www.aclweb.org/anthology/C18-1081

Akkalyoncu Yilmaz Z, Wang S, Yang W, Zhang H, Lin J (2019) Applying BERT to document retrieval with birch. 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): System Demonstrations, Association for Computational Linguistics, Hong Kong, China, pp 19–24, https://doi.org/10.18653/v1/D19-3004, https://www.aclweb.org/anthology/D19-3004

Araque O, Corcuera-Platas I, Snchez-Rada JF, Iglesias CA (2017) Enhancing deep learning sentiment analysis with ensemble techniques in social applications. Expert Syst Appl 77(C):236–246. https://doi.org/10.1016/j.eswa.2017.02.002

Barbosa L, Feng J (2010) Robust sentiment detection on Twitter from biased and noisy data. In: Proceedings of the 23rd International Conference on Computational Linguistics: Posters, Association for Computational Linguistics, pp 36–44

Bengio Y, Simard P, Frasconi P (1994) Learning long-term dependencies with gradient descent is difficult. IEEE Trans Neural Networks 5(2):157–166. https://doi.org/10.1109/72.279181

Bengio Y, Ducharme R, Vincent P (2000) A neural probabilistic language model. In: Leen T, Dietterich T, Tresp V (eds) Advances in Neural Information Processing Systems, MIT Press, vol 13, https://proceedings.neurips.cc/paper/2000/file/728f206c2a01bf572b5940d7d9a8fa4c-Paper.pdf

Bravo-Marquez F, Frank E, Mohammad SM, Pfahringer B (2016) Determining word-emotion associations from tweets by multi-label classification. In: 2016 IEEE/WIC/ACM International Conference on Web Intelligence (WI), pp 536–539, https://doi.org/10.1109/WI.2016.0091

Cambria E, Poria S, Gelbukh A, Thelwall M (2017) Sentiment analysis is a big suitcase. IEEE Intell Syst 32(6):74–80. https://doi.org/10.1109/MIS.2017.4531228

Carvalho J, Plastino A (2021) On the evaluation and combination of state-of-the-art features in twitter sentiment analysis. Artif Intell Rev 54(3):1887–1936

Chaybouti S, Saghe A, Shabou A (2021) Efficientqa : a roberta based phrase-indexed question-answering system. arxiv: 2101.02157

Chen L, Wang W, Nagarajan M, Wang S, Sheth A (2012) Extracting diverse sentiment expressions with target-dependent polarity from Twitter. In: Proceedings of the 6th International AAAI Conference on Weblogs and Social Media, pp 50–57

Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30

Devlin J, Chang MW, Lee K, Toutanova K (2019) BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, pp 4171–4186, https://doi.org/10.18653/v1/N19-1423, https://www.aclweb.org/anthology/N19-1423

Diakopoulos N, Shamma D (2010) Characterizing debate performance via aggregated Twitter sentiment. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Association for Computing Machinery, pp 1195–1198

Dodge J, Ilharco G, Schwartz R, Farhadi A, Hajishirzi H, Smith N (2020) Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. arxiv: 2002.06305

Dong L, Wei F, Tan C, Tang D, Zhou M, Xu K (2014) Adaptive recursive neural network for target-dependent Twitter sentiment classification. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: short papers, Association for Computational Linguistics, pp 49–54

Fayyad U, Piatetsky-Shapiro G, Uthurusamy R (2003) Summary from the kdd-03 panel - data mining: The next 10 years. SIGKDD Explorations 5:191–196

Felbo B, Mislove A, Søgaard A, Rahwan I, Lehmann S (2017) 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, Association for Computational Linguistics, Copenhagen, Denmark, pp 1615–1625, https://doi.org/10.18653/v1/D17-1169, https://www.aclweb.org/anthology/D17-1169

Gao Z, Feng A, Song X, Wu X (2019) Target-dependent sentiment classification with bert. IEEE Access 7:154290–154299. https://doi.org/10.1109/ACCESS.2019.2946594

Go A, Bhayani R, Huang L (2009) Twitter sentiment classification using distant supervision. Tech. Rep. CS224N, Stanford

Gonçalves P, Dalip D, Reis J, Messias J, Ribeiro F, Melo P, Araúo L, Gonçalves M, Benevenuto F (2015) Bazinga! caracterizando e detectando sarcasmo e ironia no twitter. In: Anais do IV Brazilian Workshop on Social Network Analysis and Mining, SBC, Porto Alegre, RS, Brasil, https://doi.org/10.5753/brasnam.2015.6778 https://sol.sbc.org.br/index.php/brasnam/article/view/6778

Gururangan S, Marasovic A, Swayamdipta S, Lo K, Beltagy I, Downey D, Smith NA (2020) Don't stop pretraining: Adapt language models to domains and tasks. In: Jurafsky D, Chai J, Schluter N, Tetreault JR (eds) Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020, Association for Computational Linguistics, pp 8342–8360

Howard J, Ruder S (2018) Universal language model fine-tuning for text classification. arxiv: 1801.06146

Hutto C, Gilbert E (2014) Vader: A parsimonious rule-based model for sentiment analysis of social media text. In: Proceedings of the 8th International AAAI Conference on Weblogs and Social Media

Lan Z, Chen M, Goodman S, Gimpel K, Sharma P, Soricut R (2020) Albert: A lite bert for self-supervised learning of language representations. arxiv: 1909.11942

Liu B (2020) Sentiment analysis: Mining opinions, sentiments, and emotions. Cambridge University Press

Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L, Stoyanov V (2019) Roberta: A robustly optimized bert pretraining approach. arxiv: 1907.11692

Lochter JV, Zanetti RF, Reller D, Almeida TA (2016) Short text opinion detection using ensemble of classifiers and semantic indexing. Expert Syst Appl 62:243–249

Manning CD, Raghavan P, Schütze H (2008) Introduction to Information Retrieval. Cambridge University Press, USA

Mikolov T, Sutskever I, Chen K, Corrado G, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2, NIPS'13, pp 3111–3119

Mikolov T, Grave E, Bojanowski P, Puhrsch C, Joulin A (2018) Advances in pre-training distributed word representations. In: Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)

Mohammad SM, Bravo-Marquez F, Salameh M, Kiritchenko S (2018) Semeval-2018 Task 1: Affect in tweets. In: Proceedings of 12th International Workshop on Semantic Evaluation (SemEval 2018), Association for Computational Linguistics, New Orleans, LA, USA

Nakov P, Ritter A, Rosenthal S, Sebastiani F, Stoyanov V (2016) SemEval-2016 task 4: Sentiment analysis in Twitter. In: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), Association for Computational Linguistics, San Diego, California, pp 1–18, https://doi.org/10.18653/v1/S16-1001, https://www.aclweb.org/anthology/S16-1001

Narr S, Hulfenhaus M, Albayrak S (2012) Language-independent Twitter sentiment analysis. In: Proceedings of the Workshop on Knowledge Discovery, Data Mining and Machine Learning

Nguyen DQ, Vu T, Nguyen AT (2020) Bertweet: A pre-trained language model for english tweets. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp 9–14

Pak A, Paroubek P (2010) Twitter as a corpus for sentiment analysis and opinion mining. In: Proceedings of the 7th International Conference on Language Resources and Evaluation, pp 1320–1326

Pathak AR, Agarwal B, Pandey M, Rautaray S (2020) Application of Deep Learning Approaches for Sentiment Analysis, Springer Singapore, Singapore, pp 1–31. https://doi.org/10.1007/978-981-15-1216-2_1

Pennington J, Socher R, Manning C (2014) GloVe: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Doha, Qatar, pp 1532–1543, https://doi.org/10.3115/v1/D14-1162, https://www.aclweb.org/anthology/D14-1162

Peters ME, Neumann M, Iyyer M, Gardner M, Clark C, Lee K, Zettlemoyer L (2018) Deep contextualized word representations. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), Association for Computational Linguistics, New Orleans, Louisiana, pp 2227–2237, https://doi.org/10.18653/v1/N18-1202, https://aclanthology.org/N18-1202

Petrović S, Osborne M, Lavrenko V (2010) The Edinburgh twitter corpus. In: Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics in a World of Social Media, Association for Computational Linguistics, Los Angeles, California, USA, pp 25–26, https://www.aclweb.org/anthology/W10-0513

Radford A, Narasimhan K, Salimans T, Sutskever I (2018) Improving language understanding by generative pre-training

Rosenthal S, Farra N, Nakov P (2017) SemEval-2017 task 4: Sentiment analysis in Twitter. In: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), Association for Computational Linguistics, Vancouver, Canada, pp 502–518, https://doi.org/10.18653/v1/S17-2088, https://www.aclweb.org/anthology/S17-2088

Saif H (2015) Semantic sentiment analysis of microblogs. PhD thesis, The Open University, http://oro.open.ac.uk/44063/

Saif H, Fernandez M, He Y, Alani H (2013) Evaluation datasets for Twitter sentiment analysis: a survey and a new dataset, the STS-Gold. In: Proceedings of the 1st Workshop on Emotion and Sentiment in Social and Expressive Media

Salton G, Wong A, Yang CS (1975) A vector space model for automatic indexing. Commun ACM 18(11):613–620. https://doi.org/10.1145/361219.361220

Speriosu M, Sudan N, Upadhyay S, Baldridge J (2011) Twitter polarity classification with label propagation over lexical links and the follower graph. In: Proceedings of the 1st Workshop on Unsupervised Learning in NLP, Association for Computational Linguistics, pp 53–63

Tang D, Wei F, Yang N, Zhou M, Liu T, Qin B (2014) Learning sentiment-specific word embedding for Twitter sentiment classification. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Baltimore, Maryland, pp 1555–1565, https://doi.org/10.3115/v1/P14-1146, https://www.aclweb.org/anthology/P14-1146

Thelwall M, Buckley K, Paltoglou G (2012) Sentiment strength detection for the social web. J Am Soc Inform Sci Technol 63(1):163–173

Turney PD, Pantel P (2010) From frequency to meaning: Vector space models of semantics. Journal of Artificial Intelligence Research 37(1):141–188

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. arxiv: 1706.03762

Xu P, Madotto A, Wu CS, Park JH, Fung P (2018) Emo2Vec: Learning generalized emotion representation by multi-task training. In: Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, Association for Computational Linguistics, Brussels, Belgium, pp 292–298, https://doi.org/10.18653/v1/W18-6243, https://www.aclweb.org/anthology/W18-6243

Zhu Y, Kiros R, Zemel R, Salakhutdinov R, Urtasun R, Torralba A, Fidler S (2015) Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In: Proceedings of the IEEE international conference on computer vision, pp 19–27