applied sciences

Article

Roman Urdu Sentiment Analysis Using Transfer Learning

Dun Li [1], Kanwal Ahmed [1], Zhiyun Zheng [1], Syed Agha Hassnain Mohsan [2], Mohammed H. Alsharif [3], Myriam Hadjouni [4,*], Mona M. Jamjoom [4] and Samih M. Mostafa [5]

1 School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou 450001, China
2 Ocean College, Zhejiang University, Zheda Road 1, Zhoushan 316021, China
3 Department of Electrical Engineering, College of Electronics and Information Engineering, Sejong University, Seoul 05006, Korea
4 Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia
5 Faculty of Computers and Information, South Valley University, Qena 83523, Egypt
* Correspondence: mfhaojouni@pnu.edu.sa

Abstract: Numerous studies have been conducted to meet the growing need for analytic tools capable of processing increasing amounts of textual data available online, and sentiment analysis has emerged as a frontrunner in this field. Current studies are focused on the English language, while minority languages, such as Roman Urdu, are ignored because of their complex syntax and lexical varieties. In recent years, deep neural networks have become the standard in this field. The entire potential of DL models for text SA has not yet been fully explored, despite their early success. For sentiment analysis, CNN has surpassed in accuracy, although it still has some imperfections. To begin, CNNs need a significant amount of data to train. Second, it presumes that all words have the same impact on the polarity of a statement. To fill these voids, this study proposes a CNN with an attention mechanism and transfer learning to improve SA performance. Compared to state-of-the-art methods, our proposed model appears to have achieved greater classification accuracy in experiments.

Keywords: sentiment analysis (SA); convolutional neural network (CNN); Roman Urdu language; attention mechanism; transfer learning; machine learning (ML); deep learning (DL)

Citation: Li, D.; Ahmed, K.; Zheng, Z.; Mohsan, S.A.H.; Alsharif, M.H.; Hadjouni, M.; Jamjoom, M.M.; Mostafa, S.M. Roman Urdu Sentiment Analysis Using Transfer Learning. *Appl. Sci.* **2022**, *12*, 10344. https://doi.org/10.3390/app122010344

Academic Editors: Valentino Santucci and Paolo Mengoni

Received: 17 September 2022
Accepted: 10 October 2022
Published: 14 October 2022

1. Introduction

In the subject of natural language processing (NLP), sentiment analysis is one of the most well-known subfields due to its prominence in text classification [1]. User reviews and complaints have become increasingly commonplace since the advent of social media and other user-centric platforms [2]. Automatically sorting text documents into those with the desired polarity is another consideration in the field of sentiment classification [3]. In terms of polarity, it can be either positive or negative, or even neutral. In addition to its more general meaning, sentiment analysis can be thought of as a subfield of text classification that focuses on the examination of how readers feel about and react to various things and their characteristics through the words they use to describe them. Businesses, governments, and scholars have all discovered that sentiment analysis is a useful tool for understanding public opinion, improving decision-making, and gaining corporate insight [4,5].

Academics have focused a lot of their research on SA. Compared to the English language, there are several languages, including Hindi/Roman Urdu, that are regarded as resource-poor, despite receiving less academic attention because fewer resources are available to conduct SA on them [6]. In addition to being Pakistan's national language, Urdu is spoken by around 200 million people and is written by another 100 million [3]. The Urdu language is challenging to learn and master because of its complicated alphabet, convoluted morphological structure, hazy word boundaries, and tricky stemming [7]. However, unlike other languages, Urdu lacks resources for language processing, such as stemming and lemmatization tools and stop word lists. Roman Urdu is created by combining English letters and the Urdu script. Using the English alphabet, the Urdu language can be transformed into Roman Urdu. For instance, the Roman Urdu word for "Justice" is "Insaf." Internet users in Pakistan overwhelmingly favor Roman Urdu over the traditional Nastaliq Urdu (Arabic script), according to a regional study that investigated the significance of Roman Urdu in the retrieval of information across multiple languages [8]. Despite the extensive use of Roman Urdu and Hindi, the problem of conducting SA in either of these languages has not yet been investigated in its entirety. Thus, the main objective of this research will be to explore Roman Urdu SA.

ML and DL models are widely used in various fields [9–15] and are of great success. Recent advancements in ML- and DL-based modes for SA have had a substantial effect on the efficacy of scholarly and scientific endeavors. With its impressive performance in sentiment classification, CNN has earned its place as a pivotal deep-learning model. CNNs are quite effective at sentiment analysis, yet they face several challenges [16,17]. In the first place, CNNs are limited in that they can only establish the document's polarity; they cannot offer a comprehensive comprehension of the text, for instance, identifying the determinant word for polarity classification [18]. So, even though people have brains, they still cannot highlight the most important points in a document, reducing their efficiency [19,20]. Second, CNNs are notoriously data-intensive, necessitating a sizable sample size of training data to adequately calibrate the model and a variety of finely tuned hyperparameters [21].

This study's main contribution is the development of a novel deep-learning approach for enhancing the accuracy of Roman Urdu sentiment analysis. In order to accomplish this, we have employed CNN to glean profound contextual semantics. By emulating the attention mechanism seen in human brains and prioritizing more essential words while ignoring less significant ones [22–24], we may reap the benefits of using both attention mechanisms and transfer learning. Transfer learning has helped us deal with the issue of the abundance of data sets. Training a model on big datasets in the source domain before transferring it to the target domain can increase its performance [25]. Transfer learning is an approach that avoids the enormous amount of training data that CNN requires, and this is the fundamental notion behind it. The study makes the following contributions:

- We propose a CNN with attention to Roman Urdu SA.
- We conducted a series of experiments to investigate the model's sensitivity and find the best settings for the hyper-parameters that would maximize classification accuracy.
- Transfer Learning is applied to improve small datasets.

The rest of the paper is divided into five sections. In Section 2, we will discuss the literature. The proposed methodology is laid out in great depth in Section 3. Section 4 explains the complete setup of the dataset, experimentations, and comparison of the results. In Section 5, the results and discussions are presented. Section 6 finally concludes the findings.

2. Literature Review

Various ML models have been widely employed in the past for performing sentiment analysis. In this article, we will analyze the research on DL methods such as DNN and attention-based mechanisms. Then, we will provide a quick summary of the research undertaken so far on Roman Urdu sentiment analysis.

2.1. Deep Learning-Based Sentiment Analysis

CNN, RNNs, RAEs, and DBNs are only a few of the DNNs that have been frequently employed for sentiment categorization [26] as a response to the limitations of standard machine learning methods. After applying the GRU to the recursive model and the tree structure of the LSTM, Kuta et al. [27] proposed a tree structure gated RNN. In addition, Tai et al. [28] performed sentiment analysis using an LSTM network with several complex units. They conducted more tests on a bidirectional LSTM with two layers and achieved encouraging results. Kim et al. [29] pursued a related line of questioning by evaluating a single-layer CNN extensively. They used Word2Vec vectors to train the models.

Multi-channel representation and variable-sized filters both produced the same outcomes. Zhang et al. [30] introduced an improved CNN at the character level, which was used to classify texts. Using word embedding, Yin and Schutze [31] presented a multi-channel variable-size convolutional neural network. Kalchbrenner et al. [32] developed a dynamic CNN with k-max pooling. Their model correctly processed input phrases with varying durations, and it accurately captured both short-term and long-term dependencies. Words and phrases can be represented in a matrix and vector in a tree structure, as proposed by Socher et al. [33]. Socher et al. [34] proposed Recursive Neural Tensor Network (RNTN), and it employs tensors as compositional matrices for all tree nodes. Using convolutional and RNN, Sadr et al. [35] proposed a method for sentiment classification. They employed multi-view learning to categorize the intermediate features generated by convolutional and recursive neural networks [36]. Chen et al. [37] combined Long-Short-Term Memory with a conditional random field layer to improve sentence-level sentiment analysis (BiLSTM-CRF). The simulation results on the benchmark datasets showed that their system improved phrase-level sentiment analysis. Hassan et al. [38] recommended using CNN and LSTM to infer polarity from IMDB movie reviews. This method's CNN classifier had two convolutional and two pooling layers. Shen et al. [25] used CNN and bidirectional Long-Short-Term Memory to determine movie review polarity (BLSTM). The CNN-LSTM model beat CNN and LSTM classifiers. Kamyab et al. [39] present a CNN-LSTM attention-based model (named ACL-SA). First, CNN's max-pooling is used to extract contextual features and minimize feature dimensionality. LSTM is used to capture long-term dependencies. Dashtipour et al. [40] used two deep learning algorithms, CNN and LSTM, with SVM for Persian movie reviews. Kastrati et al. [41] collected COVID-19-related comments in the Albanian language. They incorporated the attention mechanism with the DL model to capture the semantic and global context of words for SA.

DNN has produced significant results in SA; however, there is always room for improvement.

2.2. Attention-Based Network for Sentiment Analysis

Despite considerable breakthroughs in sentiment analysis, deep neural networks' performance is inadequate [26,42]. One of their typical problems is that they treat all words in phrases identically and cannot focus on critical sections [43]. The attention mechanism, with its proven ability to comprehend text effectively, has lately been utilized in a wide range of NLP applications, most notably sentiment analysis, to address this deficiency. Humans have a visual attention mechanism that helps them prioritize the information in a sentence that is most important to remember rather than encoding the entire thing. With text classification in mind, Yang et al. [19] tweaked RNN by introducing a weight to fulfill the attention function. In addition, Wang et al. [44] presented an attention-based LSTM Model that could concentrate on different sentence segments. Yuan et al. [45] suggested a domain attention model for multi-domain sentiment classification. To pick the best features relevant to each domain, their suggested model prioritized representations of those domains.

A unique approach for sentiment analysis was developed by Long et al. [24], which uses eye-tracking data to learn about the viewer's mental state. To identify the positive and negative emotions in texts, Deng et al. [46] used a sparse self-attention method to record the significance of each word. It was proposed by Gan et al. [27] to use a sparse attention-based separable dilated convolutional neural network, wherein the properties of a particular target entity are used to identify sentiment-oriented components. An organizational framework for classifying emotions and extracting topics was proposed by Pergola et al. [6]. Without using aspect-level annotation, their model extracts aspect-sentiment clusters. The authors Liang et al. [8] presented a sequence-to-sequence selective attention-based approach for the summarization of social text. They immediately optimized the ROUGE score using reinforcement learning and cross-entropy, and they employed a special gate to filter out the irrelevant data.

Despite the apparent success of using an attention mechanism on DNN, relatively some studies have investigated its effects on CNN for use in sentiment analysis.

2.3. Transfer Learning

A lack of training data is another issue for DNNs. To correctly train the model, a DNN requires a huge amount of training data; the more data that are added, the better a model will perform [45]. Transfer learning emerged due to a shortage of labeled training data. Transfer learning is utilized when the training data are too small [47]. The model is initially trained on a large dataset in the source domain before being transferred to the target domain to boost its overall performance. Transfer learning is widely used for image processing; however, sentiment analysis in NLP is limited [36]. Krizhevsky and Lee [48] studied the transferability of low-level neuronal layers across tasks. A previous study [49] examined the results of migrating deep neural network hidden layers from a larger training dataset to a smaller test dataset. Few research studies have been conducted on the possible benefits of transfer learning for sentiment analysis [19].

Tu and Zhao (2019) [50] created a model to collect source and target domain information simultaneously. It consists of a cloze task network and a convolutional hierarchical attention network to classify sentiment. Zhang et al. (2019) [28,51] introduced a method for cross-domain sentiment classification based on HAGANs. Training a generator and a discriminator alternatively produces a document representation that is both sentiment- and domain-indistinguishable. An approach for cross-domain text categorization was presented by Wang et al. (2020) [52], who combined two non-negative matrix tri-factorizations into a joint optimization framework with approximation constraints on word clusters matrices and clusters association matrices. Unsupervised domain adaptation (UDA) uses labeled source domain knowledge to comprehend unlabeled target domains [53]. Learning transferable representations from a well-labeled source domain to a comparable but unlabeled target domain, deep domain adaptation approaches have attained attractive performance [54].

2.4. Roman Urdu Sentiment Analysis

In a recent lexical normalization study for Roman Urdu [55], a phonetic technique named TERUN (Transliteration-based Encoding for Roman Urdu text Normalization) was proposed. To standardize lexically flexible terminology, TERUN utilizes Roman Hindi/Urdu linguistics. An encoder based on transliteration, a filter module, and a hash code ranker is the three interconnected components. In the encoder, a single word in Romanized Hindi or Urdu will create every hash code. The filtered hash codes are prioritized by relevancy in the third module, and the irrelevant codes are dismissed in the fourth. The purpose of this study is twofold: to standardize and categorize text.

For Roman Urdu sentiment analysis, a new term-weighting method termed discriminative feature spamming strategy (DFST) was recently presented [56]. DFST works well for sentiment analysis because it uses a term utility criterion (TUC) to pick out different words and then spams them to make them more distinctive. Experiments were run on 11,000 Roman Urdu reviews, and a specialized tokenizer was built to improve accuracy. Ayesha et al. [57] examined Roman-Urdu product reviews for SA. They experimented with Naive Bayes, Support Vector Machines, and Logistic Regression, all of which are machine learning methods. It was clear from the outcomes that SVM is a better classifier than the others. Noor et al. [58] have suggested SVM for this service. They experimented with several SVM kernels after collecting 20,000 customer reviews of products available on "Daraz.pk". The bag-of-word features and linear SVM kernel are relied upon to arrive at the claimed 60% accuracy.

A small selection of Roman Urdu reviews, including only 150 positive and 150 negative ratings, was used by Bilal et al. [59] to test their theories in 2016. The KNN, DT, and NB classifiers for machine learning were all implemented in WEKA. For all four metrics (accuracy, precision, recall, and F-measure), NB was deemed to be the clear winner over DT and KNN. In an extra effort to perform Roman Urdu SA, reviews from various websites were collected for a binary classification assignment. Ten alternative classifiers (SVM, KNN, Decision Tree, Passive Aggressive, Ensemble classifier, Perceptron, SSGD, Naive Bayes, Ridge classifier, and nearest centroid) were compared in an empirical study by Arif et al. [60]. They conducted experiments using Chi-squared, IG, and MI feature selection algorithms and three different feature representation systems (TF, TF-IDF, and Hashing vectorizer). Support vector machines fared best in an empirical comparison of several ML classifiers undertaken by the authors. Naqvi et al. [61] looked at the issue of categorizing Roman Urdu news using various methods, including Naive Bayes, Logistic Regression, Long Short-Term Memory, and Convolutional Neural Networks. The purpose of the research was to divide the news into five separate types (health, business, technology, sports, and international). When standardizing the vocabulary and cleaning up the corpus, they used a phonetic algorithm. According to the findings, Naive Bayes is the most effective classifier. Recently, Chandio et al. [62] have compared the effectiveness of their proposed SVM to that of other ML algorithms for evaluating Roman Urdu sentiment.

Researchers have used deep neural networks to classify sentiments in Roman Urdu texts. With the support of other scholars, Mehmood et al. [63] have constructed a new Roman Urdu corpus RUSA-19 that includes 10,012 reviews. The ratings come from different social media sites, and they span six categories: (i) media (television, film, and radio); (ii) food; (iii) politics and policy; (iv) mobile technology; and (v) sports. They evaluated a new dataset and proposed an RCNN-based deep network for sentiment analysis, and the proposed model was constructed using both recurrent and convolutional neural network (CNN) layers. However, numerous studies have pointed out that CNN does better with non-sequential data than with sequential data. Ghulam et al. [64] examined two deep-learning approaches for sentiment analysis in Roman Urdu to learn more about the language's performance in this area. In practice, the LSTM model and several alternative ML classifiers were explored. In a surprising turn of events, it was determined that the deep learning model outperforms its machine learning counterparts. Some recent studies [64] used a multi-channel hybrid method, proposing a strategy based on BiGRU and word embeddings. Their dataset was also insufficient, totaling only 3241 sentiment evaluations, and it is not yet available to the public. Rizwan et al. [65] studied the problem of identifying anti-Christian bigotry in Roman Urdu by employing static CNN and other word embedding methods.

Chandio et al. [62] suggested a Roman Urdu Stemmer-powered SVM. Each user review is stemmed after preprocessing. Bag-of-word transforms input text into a feature vector. SVM classifies and detects user sentiment. Chandio [66] proposes a deep recurrent architecture RU-BiLSTM for sentiment analysis in Roman Urdu. RUECD and RUSA-19 were used to test our model. Our proposed model beat baseline models in many ways, improving by 6% to 8%. Azhar, seemab [67] Huggingface's transformer models DistilBERT and XLNet outperform Logistic Regression and Nave Bayes. DistilBERT obtained 100% accuracy on our Roman Urdu dataset with only two epochs of fine-tuning, while XLNet reached 86% accuracy with four epochs. Khan et al. [68] analyzed word embeddings for Roman Urdu and English using CNN-LSTM with typical machine learning classifiers. Long-term dependency preservation (LSTM) and local feature extraction (one-layer CNN). CNN and LSTM feature maps are supplied to machine learning classifiers for final classification. Word embedding models support this. Word2Vec performed best on RUSA-19 and UCL Glove. (Qureshi et al., 2022) [69] recently suggested a SA model for natural language using roman Urdu as a case study. They built a new dataset containing 24,000 Roman Urdu sentences and used deep neural network techniques such as CNN, RNN, ID3, and Gradient Boost Trees. Logistic Regression obtained 92.25% accuracy. Sehar et al. (2021) [70] employed DL for multimodal Urdu SA utilizing Bi-LSTM and 3D-CNN. Gui et al. (2022) [71] introduced a model to predict sentiment and detect latent topics in texts; they explored topic evolution and sentiment fluctuations. Lin et al. (2022) [72] combined classic vectorization methods with their form of BERT to generate stories. These models help with adversarial training. The intensity of sentiment emotion makes sentiment analysis multiclass other than positive, negative, and neutral. Akhtar et al. (2022) [73] suggested an ensemble framework to provide more clarity on sentiment and emotion in text. Table 1 provides a summary of the state-of-the-art literature on Roman Urdu sentiment analysis.

Table 1. Summary of Existing Literature.

Classification Algorithm	Learning Method	Corpus	Accuracy
Neural Network [74]	Supervised	15,000 user reviews	0.80
NB, DT, KNN [75]	Supervised	Only 300 reviews of two classes	0.71
SVM, LR, NB, RCNN, Hybrid Multichannel Approach [64]	Supervised	3241 sentences of three classes	0.82
RCNN, Rule-based and N-gram [69]	Supervised and Unsupervised	10,021 user sentences from various websites of three classes	0.75
KNN, DT, RF, LR, NB, ANN, SVM, AdaBoost, wVoting [76]	Supervised	11,000 reviews of two classes	0.82
KNN, DT, RF, LR, NB, ANN, SVM, AdaBoost, wVoting [77]	Supervised	11,000 sentences of two classes	0.81
SVM with stemming [62]	Supervised	26,824 user reviews	0.68
CNN-LSTM [66]	Supervised	10,021 Roman Urdu user reviews, UCL contains 20,228 sentences	0.74
NB, SVM, LR, KNN, ANN, CNN, RNN, ID3 and GB Tree [67]	Supervised	24,000 reviews	0.922 LR
BiLSTM [68]	Supervised	26,824 user reviews, 10,015 user reviews	0.67
DistilBert, XLNet [69]	Supervised	21,940 Reviews	1.0, 0.862

After reviewing the existing research and examining the complications encountered, we settled on proposing a CNN that combines the benefits of transfer learning with the attention mechanism via a layer of dedicated attention neurons. In contrast to earlier research, the proposed model uses an attention mechanism following the convolutional layer to extract meaningful words from phrases. To improve performance, the suggested model makes use of transfer learning, which involves applying what is learned in one domain to other unexplored domains.

3. Methodology

In this paper, we build deep Roman Urdu SA using a DL model for Roman Urdu/Hindi sentiment analysis that integrates Word2Vec, GloVe, TF-IDF, and FastText word embedding algorithms into a CNN architecture. CNNs have a reputation as promising tools for NLP. Additionally, they enable close-by input elements to extract at lower layers while remote elements interact at higher layers, allowing them to independently regulate the duration of the dependencies. Multiple convolutional layers can thus be used to generate an abstract, hierarchical representation of input text. To get to the root of the matter, we will employ a pooling layer to glean the following info. Local features that can hold useful information are lost due to the pooling layer. To this end, the proposed model incorporates a prior attention layer to the pooling layer to not only identify the significant characteristics but also to dampen the impact of less relevant aspects and aid the pooling layer in discovering the important features considering the context. While it is true that more training data usually results in better CNN performance, we also wanted to investigate how transfer learning might affect the suggested approach.

As far as we are aware, this is the first time a deep convolutional neural network (CNN) with attention and 4channel embedding has been proposed for Roman Urdu text classification [55–73]. 4C-CNNAT was also put through a range of transfer learning tests. Figure 1 shows the framework of the proposed model.

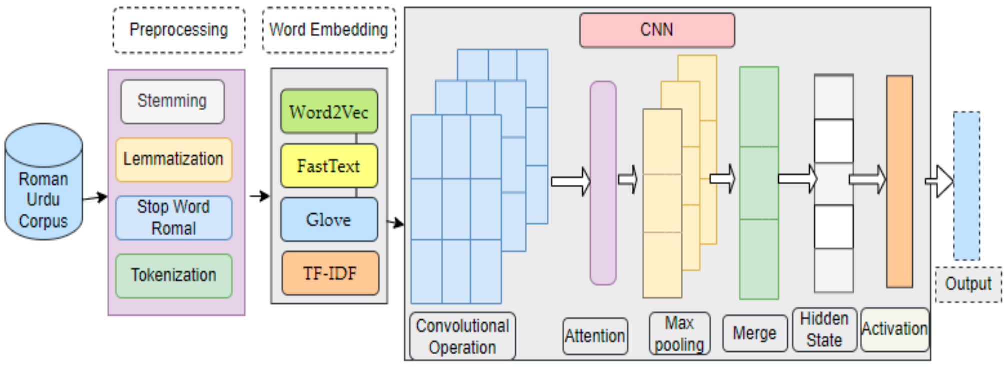


Figure 1. 4C-CNNAT for Roman Urdu Sentiment Analysis.

This section briefly explains the steps of proposed methodology.

3.1. CNN Stocked with Attention Layer and Transfer Learning

In the proposed model, we positioned the attention layer before the pooling layer in the four-layer model. Our proposed model begins with an input matrix assembled by the collaboration of word vectors derived from the input texts. After the input matrix is processed through the convolutional filters, it is possible to derive feature maps. The training concludes by merging feature maps with similar filter widths. After the CNN retrieves its features, new phrase vectors are created by extracting informative words and assigning them a higher weight. Word vectors are used as input to a fully connected network. The following is a more mathematically detailed deduction of each layer.

3.1.1. Representation Layer

When providing data to a CNN, the input must be in the form of a sentencing matrix, where each row is a word vector. With zero padding before and after the first and end words, the sentence matrix would have s dimensions if the word vector had d dimensions and the length of the phrase was $s \times d$. When the padding is set to zero, the number of times each word appears in the receptive field during convolution is constant regardless of where the word appears in the sentence. The resulting notation for the sentence matrix can be represented as this: $A \epsilon R^{s \times d}$. The input matrix was created using Word2Vec [78], Glove [79], and the FastText [80] word embedding method.

3.1.2. Convolutional Operation

To generate these novel features, a convolutional operation must be accomplished to the text matrix. Since the order of words in a sentence has a significant impact on the message it conveys, picking a filter width that is the same as the number of words in the sentence makes the most sense (d). When it comes to filters, the only parameter that can be changed is the height (h), also known as the region size.

When the convolution filter $H \epsilon R^{h \times d}$ is applied to the sentence matrix $R^{s \times d}$, the resulting sub-matrix K [i: j] is a new feature. By repeatedly applying the convolution process to the K matrix, we obtain O$\epsilon R^{s-h+1 \times d}$, as the output sequence (Equation (1)).

$$O_i = w \cdot \mathrm{K}[i : i + h - 1], \tag{1}$$

here, · is the dot product of the convolution filter's and the input submatrix's matrices, where i = 1,..., s h + 1. Each O_i is also given an activation function and a bias term $b \epsilon R$. Feature maps $F \epsilon R^{s-h+1}$ are produced at the end (Equation (2)).

$$F_i = f(O_i + b), \tag{2}$$

3.1.3. Attention Mechanism

All the words in a sentence do not contribute the same amount to the meaning of the sentence, and because the pooling layer in a CNN causes local features to be lost, there needs to be a way to draw attention to words that have a bigger effect on the meaning of a sentence when context and word interactions are taken into account.

Extracting feature maps from the same-sized filter is used to conduct the attention mechanism on the convolutional layer. Let us pretend that M distinct region widths are considered, and m distinct filters are used in the convolution layer. Therefore, a $H_{ij} \epsilon R^{h_i \times d}$ filter is applied to the sentence matrix K, where i = 1, 2,..., M, and j = 1, 2,..., m, M, to produce an m-by m feature map. This new sentence matrix $X_i \epsilon R^{nxm}$ (Equation (3)) is obtained by connecting feature maps from filters with the same size. Each cell in this matrix represents a feature that can be extracted from the input using filters of the same size. n represents the number of words.

$$X_i = \begin{bmatrix} x_{1,1} & S & x_{1,m} \\ M & O & M \\ x_{n-F_i+1,1} & K & x_{n-F_i+1,m} \end{bmatrix}, \quad (3)$$

It is the function of the attention mechanism to assign each row a relative importance for the sake of retrieving the relevant information from the text. To accomplish this, we first feed the new word matrix X_i into a single layer perceptron with the parameters $W \epsilon R^{m \times d}$ and $U_i \epsilon R^{n-h_i+1 \times d}$. This yields a hidden representation of the matrix, which we then use to predict new words (Equation (4)).

$$U_i = \tanh(X_i W + b), \quad (4)$$

To find the normalized importance weight $a_i \epsilon R^{n-h_i+1 \times 1}$, we compare each word's importance to its similarity to the context vector $u \epsilon R^{d \times 1}$ below (Equation (5)). As a process similar to that utilized by memory networks [23,33], the context vector u can be seen as a symbolic depiction of the informative words.

$$a_i = softmax(U_i u), \quad (5)$$

In particular, u starts at 0 and is learned throughout training so that each row in $\overline{X}_{ii}$ Xi is assigned the same weight. Then, we obtain a different representation of X_i by multiplying each element of $\acute{a}_i$ by the row in the X_i matrix that it corresponds to (is the element-wise product) (Equation (6)).

$$\overline{X}_i = a_i o X_i, \quad (6)$$

when the attention mechanism is applied to a representation of X_i, a new representation of X_i is created, with the informative words precisely identified. The X_i matrix is created by blending feature maps with identical filter sizes.

Then, a single-layer perceptron is used to generate a new representation of X_i (called U_i). In training, a hyper-parameter is adjusted to maximize the correlation between U_i and the content vector u, which is used to determine the normalized importance weight a_i that indicates the significance of each word. Multiplying each element of a_i by the row number in X_i yields a new representation of $\overline{X}_i$. Using the attention mechanism typically results in the extraction of more informative words.

3.1.4. Pooling and Classification

Though distinct feature maps are generated with different filter sizes, fixed-size vectors require a pooling function. To achieve this goal, several methods can be employed, including minimum pooling, maximum pooling, and the more common "average pooling." The function of the pooling layer is to minimize the number of dimensions by encapsulating the most salient aspects of each feature map. Each filter's pooling layer's output features are added together to form a feature vector O_i. Next, a fully linked Softmax layer receives the feature vector to be used as input for the desired classification. What this means is that Softmax is used to calculate the probability distribution across all sentiment categories (Equation (7)).

$$P_i = \frac{\exp(O_i)}{\sum_{j=1}^{c}\exp(O_i)}, \tag{7}$$

Using cross-entropy as the loss function, we can see how the model's distribution $\hat{P}_i(C)$ differs from the true sentiment distribution $P_i(C)$. (Equation (8)).

$$Loss = -\sum_{s\epsilon T}\sum_{i=1}^{v} P_i(C)\log(P_i(C)), \tag{8}$$

where T is used as the sample size for the test set, and v what one feels categories. The model is trained from start to finish using Stochastic Gradient Descent (SGD).

3.1.5. Transfer Learning

Transfer learning lays an emphasis on encoding and retaining knowledge in order to successfully apply what has been learned in one domain to a new, albeit related domain. Considering:

$$\begin{aligned} D_s &= \{(x_{s1}, y_{s1}), (x_{s2}, y_{s2}), \ldots, (x_{sn}, y_{sn})\} \\ D_t &= \{(x_{t1}, y_{t1}), (x_{t2}, y_{t2}), \ldots, (x_{tn}, y_{tn})\}, \end{aligned} \tag{9}$$

where D_s and D_t represents the source and target domain, Ts and Tt represent the source and target tasks, respectively. Furthermore, $x_{si}\epsilon X_s$ and $x_{ti}\epsilon X_t$ designate the i-th data in the source and the target domains, whereas $y_{si}\epsilon Y_s$ and $y_{ti}\epsilon Y_t$ display the i-th label in the source and the target domains, correspondingly. Hence, knowledge from D_s is used to improve Tt of the target prediction function ft at D_t via transfer learning. Figure 2 shows the transfer learning procedure used to train the suggested model.

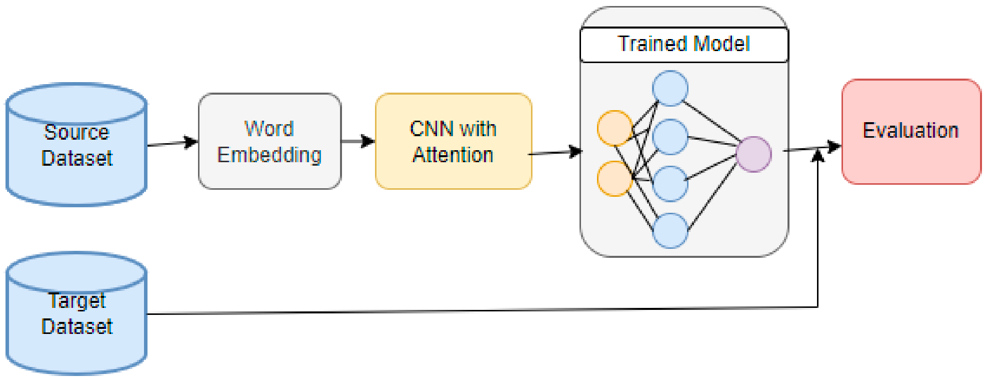


Figure 2. Proposed CNN with Attention.

Following the process flow depicted in Figure 1, the proposed model is trained on the source domain before being exported for use in the target domain. There are two distinct steps involved in conveying the trained model over. The first scenario does not subject the proposed model to any training in the desired area. This means that in the first phase (Figure 2), the model is trained on a generic domain and then tested on the target domain, whereas in the second step (Figure 3), the model is trained on the target domain to receive specific, relevant, and timely information (Figure 3). This means that the model is trained on both the source and target domains in the second process and can leverage this cross-training to improve classification accuracy. At last, this newly trained model is tested in the target domain.

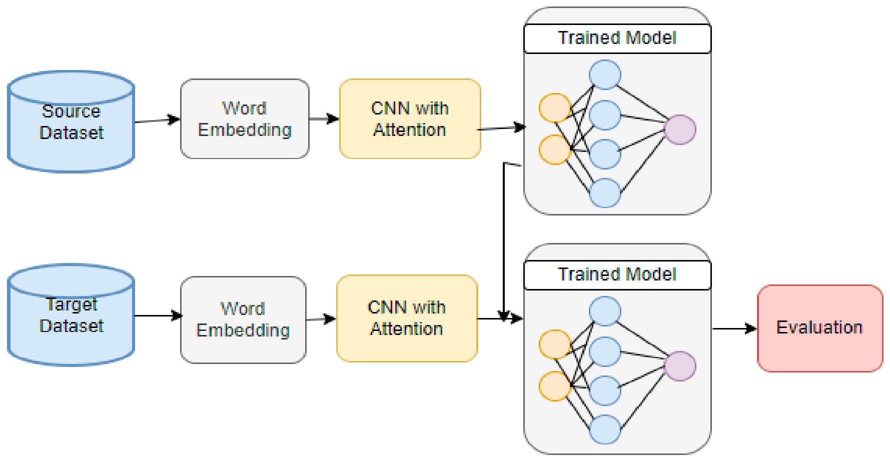


Figure 3. Learning process using transfer learning.

4. Experiments

This section includes the description of the experiments carried out to examine the effectiveness of the transfer learning model for SA. The experiments also include the measure of the accuracy of feature maps on various datasets. The sections that follow elaborate on the dataset, procedures, evaluation methods, and evaluation metrics.

4.1. Data Set

To correctly predict the performance of the purposed model, Roman Urdu standard datasets for sentiment analysis were preferred. The available datasets are RUSA 19 (https://github.com/slab-itu/rusa_19, accessed on 15 March 2022), RUDS (https://archive.ics.uci.edu/ml/datasets/Roman+Urdu+Data+Set, accessed on 15 March 2022), Roman Urdu Reviews (RUR (https://huggingface.co/datasets/roman_urdu, accessed on 15 March 2022)), DRU (https://drive.google.com/file/d/1bml7fMTjJ1ZbxaDgx-AWpJjXLOK1vGsN/view?usp= sharing, accessed on 15 March 2022), and RUSAD (https://archive.ics.uci.edu/ml/datasets/Roman+Urdu+Sentiment+Analysis+Dataset+%28RUSAD%29, accessed on 15 March 2022). We divided the datasets into two categories, i.e., the source and target domain. This division is based on the size of the dataset, as small size data set are not enough to train a model, so it is chosen as the target domain for transfer learning. The details of the datasets, along with the number of data, are presented in Table 2.

Table 2. Dataset Statistics.

Category	Dataset	Total Instances	Positive	Negative	Neutral
Source Domain	RUDS [81]	20,229	6015	5287	8927
	RUR [75]	15,677	4987	6023	4667
	DRU [69]	24,000	12,000	12,000	-
Target Domain	RUSA 19 [63]	10,021	3778	2941	3302
	RUSAD [55,76]	11,000	5686	5314	-

4.2. Evaluation Measures

To evaluate the efficacy of our classifier, we compute the accuracy, Recall, F1-score, and precision.

$$Percision = \frac{TP}{TP + FP}, \quad (10)$$

$$Recall = \frac{TP}{TP + FN}, \quad (11)$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \quad (12)$$

$$F - Measure = 2 \times \frac{Percision \times Recall}{Percision + Recall}, \quad (13)$$

where *TP* = True Positive, *TN* = true Negative, *FP* = False Positive, *FN* = False Negative.

4.3. Model Configuration

The python TensorFlow karas DL library was used in our experiments. On top of that, sckit-learn was used to randomly divide the data for use in both the training and testing phases. The training data require pre-processing data. Several state-of-the-art techniques were utilized for this task, including Stanford core NLP used for tokenization, Word2Vec [78], Glove [79], and FastText [80] models used to form word embeddings. In the training process of word embeddings, the dimension of the vector was attuned to 200, and the window size was set to 3. The word embedding vector was updated using ADADELTA along with a learning rate of 0.01. As the filter size was considered a hyperparameter, it resulted in filter sizes 3, 4, and 5, along with 128 filters providing encouraging results. The dropout rate in the convolution layer was regularized to 0.5. Softmax was used as the activation function because it smooths sharply and converges faster than other activation functions in vogue. The batch size was set to 32. The model was trained using 50 epochs. After each round of training, the validation accuracy was also calculated. If the training accuracy was discovered to be dropping, the process was terminated immediately.

5. Results and Discussions

The proposed model was tested in five discrete configurations to achieve the best possible outcome.

CNN-A with Random Input Vector (RI-CNNA): As input, it employs randomly initialized vectors.

CNN-A with Static Input Vector (SI- CNNA): As input, it takes Word2Vec-obtained pre-trained word vectors. There is no iterative updating of weights during the training period.

CNN-A with Non-Static-Input Vector (NS-CNNA): Word2Vec's pre-trained word vectors are used as the input. During training, weights are continuously adjusted.

CNN-A with 2Channel- Input Vector (2C-CNNAT): Word2Vec's pre-trained word vectors are used with randomly initialized vectors to train the model.

CNN-A with 4Channel- Input Vector (4C-CNNAT): A mixture of Random, Word2Vec, Glove, and FastText's pre-trained word vectors are used as input. Throughout the course of training, the weights are also adjusted.

The proposed model was only trained on the target datasets to create a foundation for equitable correspondence between the proposed model and other ML and DL techniques. The output was measured in the form of accuracy, precision, recall, and f1 score matrices. The detailed comparison of the outputs for each test is presented in Tables 3 and 4.

It is evident that the proposed model, in its various forms, outperforms the state-of-the-art by a small margin and is thus a viable option for transfer learning. Using randomly initialized vectors as an input may be why RI-CNNA has the worst classification performance among the presented model variants on both target datasets. The use of pre-trained vectors that are able to address the semantic sparsity problem has been linked to improved performance across a number of variants. Further, as SI-CNNA has the second-lowest classification accuracy after RI-CNNA, it is evident that updating word vectors during training could result in a greater performance regardless of whether the word vectors were previously trained or not. In conclusion, 4C-CNNA achieves 0.853% accuracy in classification on the RUSA-19 dataset and 0.903% accuracy on the RUSAD dataset. As seen in the results, the proposed model, along with its distinct configurations, performs better than the existing techniques. Therefore, it appeared to provide a better performance in the case of transfer learning.

Table 3. 4C-CNNAT performance comparison with various ML/DL classifiers using RUSA-19 corpus.

Classifiers	Precision	Recall	F1 Score	Accuracy
RF [82]	0.725	0.726	0.725	0.716
LR [83]	0.729	0.730	0.729	0.698
SVM [58]	0.746	0.761	0.753	0.744
NB [84]	0.725	0.731	0.727	0.712
KNN [59]	0.708	0.719	0.713	0.696
LSTM [64]	0.783	0.772	0.777	0.774
RCNN [63]	0.751	0.738	0.744	0.700
CNN [65]	0.742	0.741	0.741	0.732
RI-CNNA	0.803	0.783	0.792	0.792
SI- CNNA	0.824	0.822	0.822	0.802
NS-CNNA	0.829	0.825	0.826	0.813
2C-CNNAT	0.841	0.841	0.841	0.847
4C-CNNAT	0.851	0.854	0.852	0.853

Table 4. 4C-CNNAT performance comparison with various ML/DL classifiers using RUSAD corpus.

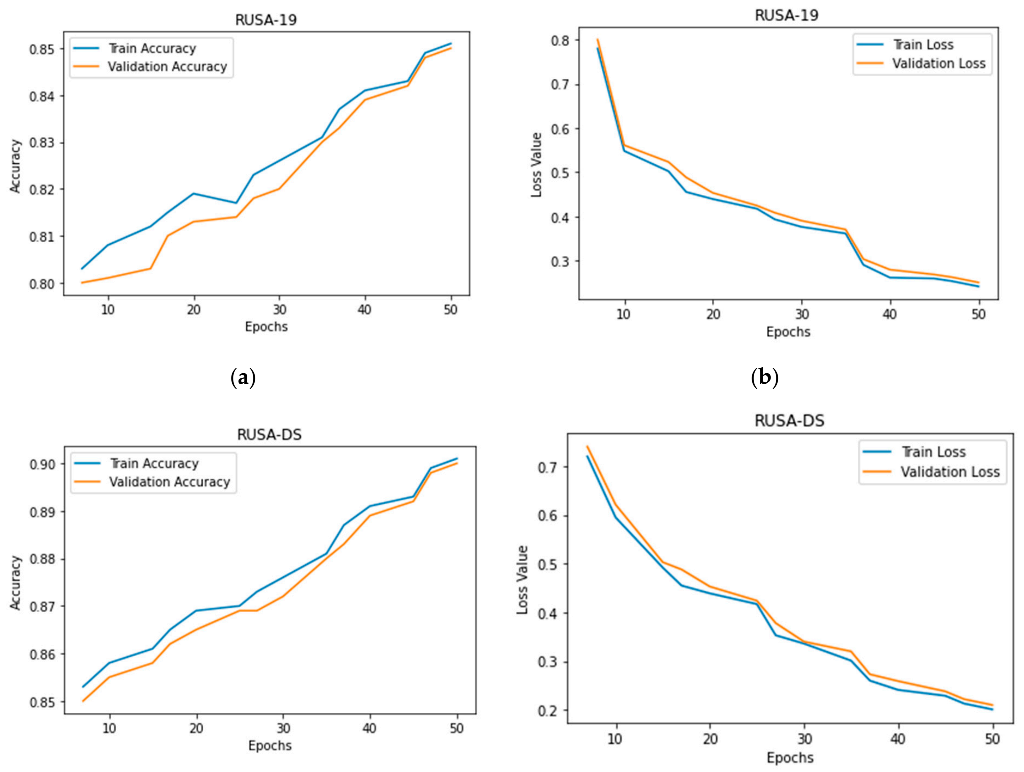
Classifiers	Precision	Recall	F1 Score	Accuracy
RF [82]	0.770	0.770	0.770	0.763
LR [83]	0.760	0.762	0.760	0.759
SVM [58]	0.841	0.850	0.845	0.844
NB [84]	0.811	0.812	0.811	0.805
KNN [59]	0.789	0.792	0.790	0.787
LSTM [64]	0.843	0.872	0.857	0.864
RCNN [63]	0.751	0.738	0.744	0.700
CNN [65]	0.755	0.753	0.753	0.748
RI-CNNA	0.853	0.853	0.853	0.852
SI- CNNA	0.859	0.842	0.850	0.852
NS-CNNA	0.861	0.865	0.862	0.863
2C-CNNAT	0.871	0.861	0.865	0.877
4C-CNNAT	0.901	0.894	0.897	0.903

Figure 4 depicts a plot of classification accuracy versus loss values per epoch throughout training and validation sets for 4C-CNNA- on the RUSA-19 and RUSAD. As can be seen in the figure, as the number of epochs is raised, classification accuracy increases while the loss value decreases.

Moreover, there was a point of stability in the validation loss plot, and the validation loss was only slightly lower than the training loss. In light of this, we can say that the model has been appropriately adjusted and provides a decent fit.

Effect of Transfer Learning

One of the disadvantages of CNNs is their variable number of hyper-parameters, which necessitates practitioners to define the precise model architecture. Considering that the values of the hyper-parameters have a substantial impact on the performance of DNN, we chose to improve the proposed model's hyper-parameters on the source domains and then apply the optimal parameters to the target domains while employing transfer learning. The prior mentioned foundation for equitable correspondence was utilized in the experimentation of the transfer learning process. Then, 10% of the training data, along with ten-fold cross-validation, was utilized as the test set. Each experiment was repeated five times with identical parameters to achieve stable and accurate results.



(**a**) (**b**)

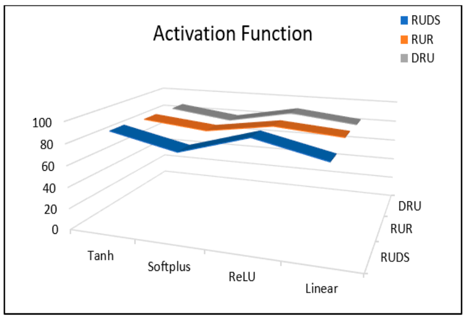
(**c**) (**d**)

Figure 4. 4C-CNNA evaluation on RUSA-19 and RUSA dataset: (**a**) classification accuracy of RUSA-19 on training and validation samples; (**b**) cross-entropy loss on RUSA-19 for training and validation samples (**c**) classification accuracy of RUSAD on training and validation samples; (**d**) cross-entropy loss on RUSAD for training and validation samples.

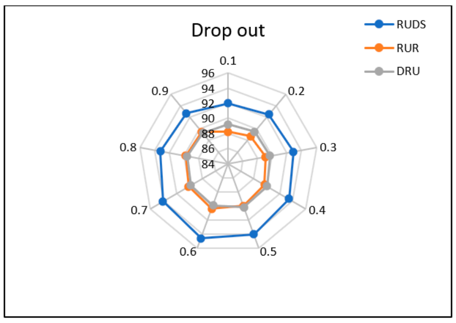
All of the parameters were kept constant except the filter size to emphasize the effects of filter size in the experimentation process. Consequently, filters (4,5,6) demonstrated a successful outcome for all datasets. Similarly, all of the parameters were kept constant except for the number of filters, to emphasize the effects of the number of filters in some experiments. During experimentation, the number of filters was attuned to 300 on DRU and RUDS, along with a dropout rate of 0.6, to achieve the highest classification accuracy. However, the optimal number of filters to achieve maximal accuracy on DRU was 128, along with a dropout rate of 0.5. A number of activation functions were utilized to achieve the finest outcomes, including SoftMax, ReLU, SoftPlus, and Tanh, but ReLU surpassed the performance of all other activation functions due to its faster convergence rate. Figure 5a shows the accuracy of the source dataset with different activation functions. Figure 5b shows the change in accuracy with different dropout rates.

Tables 5–8 shows the performance of proposed model in comparison to the baseline ML/DL models.

To compare the results of the learning process (Figure 2), firstly, the proposed model was employed for sentiment analysis without prior training on the target domain. In the second phase, using the above-mentioned attuned configurations, the proposed model was incrementally trained on the target domain. The comparison of the two phases of the transfer learning process, along with the accuracy, is shown in Table 8.



(**a**)



(**b**)

Figure 5. (**a**) classification accuracy on source datasets with different activation functions; (**b**) classification accuracy on source datasets with different Dropout values.

Table 5. 4C-CNNAT performance comparison with various ML/DL classifiers using RUDS corpus.

Classifiers	Precision	Recall	F1 Score	Accuracy
RF [82]	0.712	0.725	0.690	0.707
LR [83]	0.706	0.700	0.695	0.607
SVM [58]	0.740	0.750	0.714	0.731
NB [84]	0.716	0.731	0.690	0.709
KNN [59]	0.710	0.714	0.670	0.791
LSTM [64]	0.783	0.782	0.797	0.774
RCNN [63]	0.751	0.733	0.754	0.720
CNN [65]	0.759	0.756	0.753	0.743
4C-CNNAT	0.881	0.884	0.887	0.903

Table 6. 4C-CNNAT performance comparison with various ML/DL classifiers using RUR corpus.

Classifiers	Precision	Recall	F1 Score	Accuracy
RF [82]	0.780	0.780	0.780	0.773
LR [83]	0.770	0.772	0.770	0.769
SVM [58]	0.851	0.860	0.855	0.854
NB [84]	0.821	0.822	0.821	0.815
KNN [59]	0.809	0.802	0.800	0.797
LSTM [64]	0.873	0.862	0.867	0.864
RCNN [63]	0.831	0.838	0.824	0.820
CNN [65]	0.825	0.823	0.823	0.818
4C-CNNAT	0.911	0.914	0.917	0.908

As it is clear from the results, the accuracy of the proposed model is lower when it is used directly on the target domain without training. This only signifies that source domain knowledge is not entirely enough to achieve high accuracy on the target domain. In the second phase, the performance and accuracy of the proposed model increased significantly with the use of incremental training on the target domain.

The sizable range of the source domains, along with fine word embeddings, also provided support for learning more contextual information. All of the factors combined resulted in the proposed model remarkably improving the accuracy of the classification process. The proposed model reflects different classification accuracy of source domains, and this indicates that in the transfer learning process, choosing a source domain corresponding to a target domain is an essential task. Some datasets have highly unmatched classification variables, which result in lower accuracy. However, the size and vocabulary metrics of either source or target dataset can also affect the performance. We also concatenated the source datasets to train the model and evaluated the target datasets. Table 9 presents the accuracy obtained on the target dataset using transfer learning from the concatenated source dataset.

Table 7. 4C-CNNAT performance comparison with various ML/DL classifiers using DRU corpus.

Classifiers	Precision	Recall	F1 Score	Accuracy
RF [82]	0.791	0.791	0.781	0.793
LR [83]	0.782	0.774	0.772	0.769
SVM [58]	0.852	0.861	0.856	0.852
NB [84]	0.831	0.832	0.831	0.817
KNN [59]	0.807	0.802	0.800	0.799
LSTM [64]	0.873	0.872	0.867	0.864
RCNN [63]	0.851	0.858	0.854	0.840
CNN [65]	0.866	0.864	0.855	0.858
4C-CNNAT	0.923	0.916	0.919	0.915

Table 8. Accuracy of the proposed model with transfer learning on target domains.

Proposed Model	Description	Accuracy
Without incremental training	RUDS-> RUSA-19	0.778
With incremental training	RUDS->RUSA-19	0.946
Without incremental learning	RUR->RUSA-19	0.682
With incremental learning	RUR->RUSA-19	0.884
Without incremental learning	DRU->RUSA-19	0.763
With incremental learning	DRU->RUSA-19	0.872
Without incremental training	RUDS-> RUSAD	0.808
With incremental training	RUDS-> RUSAD	0.946
Without incremental learning	RUR-> RUSAD	0.782
With incremental learning	RUR-> RUSAD	0.904
Without incremental learning	DRU-> RUSAD	0.793
With incremental learning	DRU-> RUSAD	0.902

Table 9. Accuracy of target datasets with transfer learning from training the model on concatenated source datasets.

Target Dataset	Accuracy
RUSA-19	0.972
RUSAD	0.963

As it is obvious that the proposed model achieved higher accuracy on the target datasets when trained on the merged datasets of the source domain. The reason for this higher accuracy could be that merging different datasets increases the domain similarity of the source and target domain.

We have compared the proposed model's performance on the target datasets. 4C-CNNAT achieved higher accuracy on target datasets as compared to the state-of-the-art models. We trained the proposed model on the source datasets and transferred learning on each dataset and compared the accuracy. The major finding indicates that when employed directly on the target domain without training, the suggested model's accuracy is reduced. This simply means that the source domain knowledge is insufficient to achieve high accuracy on the target domain. With the application of incremental training on the target domain in the second phase, the performance and accuracy of the intended model improved dramatically.

6. Conclusions

The goal of SA is to learn about the public's sentiments by analyzing data collected from various social sources. In Roman Urdu, SA is challenging because of semantic and syntactic restrictions and the input sentence's terms interdependence. Consequently, this research was conducted to establish a mechanism to determine people's feelings towards a particular matter based on their Roman Urdu evaluations. This study's intended audience is the inhabitants of the subcontinent. Urdu, written in roman script, was the target language. This study contributes in two ways. With the objective of sentiment categorization in mind, a CNN with an attention mechanism before the pooling layer was proposed. This neural network provides weight to the most crucial parts of sentences and uses context to determine the polarity of the words. Since a shortage of training data is widely recognized as one of deep learning's biggest obstacles, we next investigated the impact of transfer learning and the sensitivity of the recommended model parameters. The empirical results showed that the proposed attention-based CNN significantly outperformed state-of-the-art alternatives. The accuracy of the classifications was also greatly improved using transfer learning. This work paves the way for additional research to enhance the category.

Author Contributions: Conceptualization, D.L.; Data curation, K.A., Z.Z. and S.A.H.M.; Formal analysis, K.A., S.A.H.M., M.H.A. and S.M.M.; Funding acquisition, M.H. and M.M.J.; Investigation, Z.Z., M.H.A., M.H., M.M.J. and S.M.M.; Methodology, D.L. and S.M.M.; Project administration, M.M.J. and S.M.M.; Resources, Z.Z., M.H.A. and M.M.J.; Software, K.A. and M.H.; Validation, M.H. and S.M.M.; Writing—original draft, D.L. and S.M.M.; Writing—review and editing, M.H., M.M.J. and S.M.M. All authors have read and agreed to the published version of the manuscript.

Funding: Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R104), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R104), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ligthart, A.; Catal, C.; Tekinerdogan, B. Systematic reviews in sentiment analysis: A tertiary study. *Artif. Intell. Rev.* **2021**, *54*, 4997–5053. [CrossRef]
2. Imran, A.S.; Daudpota, S.M.; Kastrati, Z.; Batra, R. Cross-cultural polarity and emotion detection using sentiment analysis and deep learning on COVID-19 related tweets. *IEEE Access* **2020**, *8*, 181074–181090. [CrossRef] [PubMed]
3. Liu, W.; Wang, Z.; Liu, X.; Zeng, N.; Liu, Y.; Alsaadi, F.E. A survey of deep neural network architectures and their applications. *Neurocomputing* **2017**, *234*, 11–26. [CrossRef]
4. Birjali, M.; Kasri, M.; Beni-Hssane, A. A comprehensive survey on sentiment analysis: Approaches, challenges and trends. *Knowl. Based Syst.* **2021**, *226*, 107134. [CrossRef]
5. Kastrati, Z.; Dalipi, F.; Imran, A.S.; Nuci, K.P.; Wani, M.A. Sentiment Analysis of Students' Feedback with NLP and Deep Learning: A Systematic Mapping Study. *Appl. Sci.* **2021**, *11*, 3986. [CrossRef]
6. Pergola, G.A.; Gui, L.; He, Y. TDAM: A topic-dependent attention model for sentiment analysis. *Inf. Process. Manag.* **2019**, *56*, 102084. [CrossRef]
7. Du, H.; Xu, X.; Cheng, X.; Wu, D.; Liu, Y.; Yu, Z. Aspect-specific sentimental word embedding for sentiment analysis of online reviews. In Proceedings of the 25th International Conference Companion on World Wide Web, International World Wide Web Conferences Steering Committee, Montreal, QC, Canada, 11–15 April 2016; pp. 29–30.
8. Liang, Z.; Du, J.; Li, C. Abstractive Social Media Text Summarization using Selective Reinforced Seq2Seq Attention Model. *Neurocomputing* **2020**, *410*, 432–440. [CrossRef]

9. Luo, G.; Yuan, Q.; Li, J.; Wang, S.; Yang, F. Artificial intelligence powered mobile networks: From cognition to decision. *IEEE Netw.* **2022**, *36*, 136–144. [CrossRef]
10. Liao, L.; Du, L.; Guo, Y. Semi-Supervised SAR Target Detection Based on an Improved Faster R-CNN. *Remote Sens.* **2021**, *14*, 143. [CrossRef]
11. Li, J.; Xu, K.; Chaudhuri, S.; Yumer, E.; Zhang, H.; Guibas, L. GRASS: Generative recursive autoencoders for shape structures. *ACM Trans. Graph.* **2017**, *36*, 1–14. [CrossRef]
12. Zhang, J.; Zhu, C.; Zheng, L.; Xu, K. ROSEFusion: Random optimization for online dense reconstruction under fast camera motion. *ACM Trans. Graph.* **2021**, *40*, 1–17. [CrossRef]
13. Zhao, H.; Zhu, C.; Xu, X.; Huang, H.; Xu, K. Learning practically feasible policies for online 3D bin packing. *Sci. China Inf. Sci.* **2021**, *65*, 1–17. [CrossRef]
14. Lin, Z.; Wang, H.; Li, S. Pavement anomaly detection based on transformer and self-supervised learning. *Autom. Constr.* **2022**, *143*, 104544. [CrossRef]
15. Xiong, Z.; Weng, X.; Wei, Y. SandplayAR: Evaluation of psychometric game for people with generalized anxiety disorder. *Arts Psychother.* **2022**, *80*, 101934. [CrossRef]
16. Ashraf, M.; Khan, L.; Tahir, M.; Alghamdi, A.; Alqarni, M.; Sabbah, T.; Khan, M. A study on usability awareness in local IT industry. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 427–432. [CrossRef]
17. Xie, X.; Ge, S.; Hu, F.; Xie, M.; Jiang, N. An improved algorithm for sentiment analysis based on maximum entropy. *Soft Comput.* **2019**, *23*, 599–611. [CrossRef]
18. Sadr, H.; Soleimandarabi, M.N.; Pedram, M.; Teshnelab, M. Unified Topic-Based Semantic Models: A Study in Computing the Semantic Relatedness of Geographic Terms. In Proceedings of the 2019 5th International Conference on Web Research (ICWR), Tehran, Iran, 24–25 April 2019; pp. 134–140.
19. Yang, Z.; Yang, D.; Dyer, C.; He, X.; Smola, A.; Hovy, E. Hierarchical attention networks for document classification. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics, Human Language Technologies, San Diego, CA, USA; 2016; pp. 1480–1489.
20. Zhang, Z.; Zou, Y.; Gan, C. Textual sentiment analysis via three different attention convolutional neural networks and cross-modality consistent regression. *Neurocomputing* **2018**, *275*, 1407–1415. [CrossRef]
21. Sadr, H.; Pedram, M.M.; Teshnelab, M. Improving the Performance of Text Sentiment Analysis using Deep Convolutional Neural Network Integrated with Hierarchical Attention Layer. *Int. J. Inf. Commun. Technol. Res.* **2019**, *11*, 57–67.
22. Liu, R.; Shi, Y.; Ji, C.; Jia, M. A survey of sentiment analysis based on transfer learning. *IEEE Access* **2019**, *7*, 85401–85412. [CrossRef]
23. Sukhbaatar, S.; Weston, J.; Fergus, R. End-to-end memory networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 2440–2448.
24. Long, Y.; Xiang, R.; Lu, Q.; Huang, C.-R.; Li, M. Improving attention model based on cognition grounded data for sentiment analysis. *IEEE Trans. Affect. Comput. Early Access* **2019**, *12*, 900–912. [CrossRef]
25. Shen, Q.; Wang, Z.; Sun, Y. Sentiment Analysis of Movie Reviews Based on CNN-BLSTM. In Proceedings of the International Conference on Intelligence Science, Durgapur, India, 24–27 February 2021; Springer: Berlin/Heidelberg, Germany, 2017; pp. 164–171.
26. Yadav, A.; Vishwakarma, D.K. Sentiment analysis using deep learning architectures: A review. *Artif. Intell. Rev.* **2020**, *53*, 4335–4385. [CrossRef]
27. Gan, C.; Wang, L.; Zhang, Z.; Wang, Z. Sparse attention based separable dilated convolutional neural network for targeted sentiment analysis. *Knowl. Based Syst.* **2020**, *188*, 104827. [CrossRef]
28. Zhang, K.; Zhang, H.; Liu, Q.; Zhao, H.; Zhu, H.; Chen, E. Interactive attention transfer network for cross-domain sentiment classification. In Proceedings of the 33rd AAAI Conference on Artificial intelligence, Hilton Hawaiian Village, Honolulu, HI, USA, 27 January–1 February 2019; pp. 5773–5780.
29. Tai, K.S.; Socher, R.; Manning, C.D. Improved semantic representations from tree-structured long shortterm memory networks. *arXiv* **2015**, arXiv:1503.00075.
30. Kim, Y. Convolutional neural networks for sentence classification. *arXiv* **2014**, arXiv:1408.5882.
31. Zhang, X.; Zhao, J.; LeCun, Y. Character-level convolutional networks for text classification. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 649–657.
32. Yin, W.; Schütze, H.; Xiang, B.; Zhou, B. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *arXiv* **2015**, arXiv:1512.05193.
33. Kalchbrenner, N.; Grefenstette, E.; Blunsom, P. A convolutional neural network for modelling sentences. *arXiv* **2014**, arXiv:1404.2188.
34. Socher, R.; Huval, B.; Manning, C.D.; Ng, A.Y. Semantic Compositionality through Recursive MatrixVector Spaces. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Jeju Island, Korea, 12–14 July 2012.
35. Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C.D.; Ng, A.; Potts, C. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, WA, USA, 19–21 October 2013.

36. Sadr, H.; Pedram, M.M.; Teshnehlab, M. A Robust Sentiment Analysis Method based on Sequential Combination of Convolutional and Recursive Neural Networks. *Neural Process. Lett.* **2019**, *50*, 1–17. [CrossRef]
37. Chen, G.; Ye, D.; Xing, Z.; Chen, J.; Cambria, E. Ensemble application of convolutional and recurrent neural networks for multi-label text categorization. In Proceedings of the International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, USA, 14–19 May 2017; pp. 2377–2383.
38. Hassan, A.; Mahmood, A. Deep Learning approach for sentiment analysis of short texts. In Proceedings of the 2017 3rd International Conference on Control, Automation and Robotics (ICCAR), Nagoya, Japan, 24–26 April 2017; pp. 705–710.
39. Kamyab, A.; Liu, G.; Adjeisah, M. Attention-based CNN and Bi-LSTM model based on TF-IDF and glove word embedding for sentiment analysis. *Appl. Sci.* **2021**, *11*, 11255. [CrossRef]
40. Dashtipour, K.; Gogate, M.; Adeel, A.; Larijani, H.; Hussain, A. Sentiment analysis of persian movie reviews using deep learning. *Entropy* **2021**, *23*, 596. [CrossRef]
41. Kastrati, Z.; Ahmedi, L.; Kurti, A.; Kadriu, F.; Murtezaj, D.; Gashi, F. A deep learning sentiment analyser for social media comments in low-resource languages. *Electronics* **2021**, *10*, 1133. [CrossRef]
42. Pang, B.; Lee, L. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, Stroudsburg, PA, USA, 25–30 June 2005; pp. 115–124.
43. Wadawadagi, R.; Pagi, V. Sentiment analysis with deep neural networks: Comparative study and performance assessment. *Artif. Intell. Rev.* **2020**, *53*, 6155–6195. [CrossRef]
44. Wang, Y.; Huang, M.; Zhao, L. Attention-based LSTM for aspect-level sentiment classification. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TX, USA, 1–5 November 2016; pp. 606–615.
45. Yuan, Z.; Wu, S.; Wu, F.; Liu, J.; Huang, Y. Domain attention model for multi-domain sentiment classification. *Knowl. Based Syst.* **2018**, *155*, 1–10. [CrossRef]
46. Deng, D.; Jing, L.; Yu, J.; Sun, S. Sparse self-attention LSTM for sentiment lexicon construction. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2019**, *27*, 1777–1790. [CrossRef]
47. Semwal, T.; Yenigalla, P.; Mathur, G.; Nair, S.B. A practitioners' guide to transfer learning for text classification using convolutional neural networks. In Proceedings of the 2018 SIAM International Conference on Data Mining, San Diego, CA, USA, 3–5 May 2018; pp. 513–521.
48. Zhuang, F.; Qi, Z.; Duan, K.; Xi, D.; Zhu, Y.; Zhu, H.; Xiong, H.; He, Q. A comprehensive survey on transfer learning. *arXiv* **2019**, arXiv:1911.02685. [CrossRef]
49. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
50. Tu, M.; Zhao, X. CCHAN: An end to end model for cross domain sentiment classification. *IEEE Access* **2019**, *7*, 50232–50239. [CrossRef]
51. Zhang, Y.; Miao, D.; Wang, J. Hierarchical attention generative adversarial networks for cross-domain sentiment classification. *arXiv preprint* **2019**, arXiv:1903.11334.
52. Wang, D.; Lu, C.; Wu, J.; Liu, H.; Zhang, W.; Zhuang, F.; Zhang, H. Softly associative transfer learning for cross-domain classification. *IEEE Trans. Cybern.* **2020**, *50*, 4709–4721. [CrossRef] [PubMed]
53. Xie, B.; Li, S.; Lv, F.; Liu, C.H.; Wang, G.; Wu, D. A Collaborative Alignment Framework of Transferable Knowledge Extraction for Unsupervised Domain Adaptation. *IEEE Trans. Knowl. Data Eng.* **2022**, *1*. [CrossRef]
54. Li, S.; Liu, C.H.; Lin, Q.; Wen, Q.; Su, L.; Huang, G.; Ding, Z. Deep Residual Correction Network for Partial Domain Adaptation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 2329–2344. [CrossRef]
55. Mehmood, K.; Essam, D.; Shafi, K.; Malik, M. An unsupervised lexical normalization for Roman Hindi and Urdu sentiment analysis. *Inf. Process. Manag.* **2020**, *57*, 102368. [CrossRef]
56. Mehmood, K.; Essam, D.; Shafi, K.; Malik, M.K. Discriminative Feature Spamming Technique for Roman Urdu Sentiment Analysis. *IEEE Access* **2019**, *7*, 47991–48002. [CrossRef]
57. Rafique, A.; Malik, M.K.; Nawaz, Z.; Bukhari, F.; Jalbani, A.H. Sentiment analysis for roman Urdu. *Mehran Univ. Res. J. Eng. Technol.* **2019**, *38*, 463–470. [CrossRef]
58. Noor, F.; Bakhtyar, M.; Baber, J. Sentiment analysis in E-commerce using SVM on roman Urdu text. In Proceedings of the International Conference for Emerging Technologies in Computing, London, UK, 19–20 August 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 213–222.
59. Bilal, M.; Israr, H.; Shahid, M.; Khan, A. Sentiment classification of Roman-Urdu opinions using Nave Bayesian, decision tree and KNN classification techniques. *J. King Saud Univers. Comput. Inf. Sci.* **2016**, *28*, 330–344. [CrossRef]
60. Arif, H.; Munir, K.; Danyal, A.S.; Salman, A.; Fraz, M.M. Sentiment analysis of roman urdu/hindi using supervised methods. *Proc. ICICC* **2016**, *8*, 48–53.
61. Naqvi, R.A.; Khan, M.A.; Malik, N.; Saqib, S.; Alyas, T.; Hussain, D. Roman Urdu news headline classification empowered with machine learning. *Comput. Mater. Contin.* **2020**, *65*, 1221–1236.
62. Chandio, B.; Shaikh, A.; Bakhtyar, M.; Alrizq, M.; Baber, J.; Sulaiman, A.; Rajab, A.; Noor, W. Sentiment Analysis of Roman Urdu on E-Commerce Reviews Using Machine Learning. *CMES-Comput. Model. Eng. Sci.* **2022**, *131*, 1263–1287. [CrossRef]

63. Mahmood, Z.; Safder, I.; Nawab, R.M.A.; Bukhari, F.; Nawaz, R.; Alfakeeh, A.S.; Aljohani, N.R.; Hassan, S.U. Deep sentiments in Roman Urdu text using Recurrent Convolutional Neural Network model. *Inf. Process. Manag.* **2020**, *57*, 102233. [CrossRef]
64. Ghulam, H.; Zeng, F.; Li, W.; Xiao, Y. Deep learning-based sentiment analysis for roman urdu text. *Procedia Comput. Sci.* **2019**, *147*, 131–135. [CrossRef]
65. Rizwan, H.; Shakeel, M.H.; Karim, A. Hate-speech and offensive language detection in roman Urdu. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16–20 November 2020; pp. 2512–2522.
66. Chandio, B.A.; Imran, A.S.; Bakhtiar, M.; Daudpota, S.M.; Baber, J. Attention-Based RU-BiLSTM Sentiment Analysis Model for Roman Urdu. *Appl. Sci.* **2022**, *12*, 3641. [CrossRef]
67. Azhar, N.; Latif, S. Roman Urdu Sentiment Analysis Using Pre-trained DistilBERT and XLNet. In Proceedings of the 2022 Fifth International Conference of Women in Data Science at Prince Sultan University (WiDS PSU), Riyadh, Saudi Arabia, 28–29 March 2022; pp. 75–78.
68. Khan, L.; Amjad, A.; Afaq, K.M.; Chang, H.-T. Deep Sentiment Analysis Using CNN-LSTM Architecture of English and Roman Urdu Text Shared in Social Media. *Appl. Sci.* **2022**, *12*, 2694. [CrossRef]
69. Qureshi, M.A.; Asif, M.; Hassan, M.F.; Abid, A.; Kamal, A.; Safdar, S.; Akber, R. Sentiment analysis of reviews in natural language: Roman Urdu as a case study. *IEEE Access* **2022**, *10*, 24945–24954. [CrossRef]
70. Sehar, U.; Kanwal, S.; Dashtipur, K.; Mir, U.; Abbasi, U.; Khan, F. Urdu Sentiment Analysis via Multimodal Data Mining Based on Deep Learning Algorithms. *IEEE Access* **2021**, *9*, 153072–153082. [CrossRef]
71. Gui, L.; Leng, J.; Zhou, J.; Xu, R.; He, Y. Multi task mutual learning for joint sentiment classification and topic detection. *IEEE Trans. Knowl. Data Eng.* **2022**, *34*, 1915–1927. [CrossRef]
72. Lin, J.-W.; Chang, R.-G. Chinese story generation of sentence format control based on multi-channel word embedding and novel data format. *Soft Comput.* **2022**, *26*, 2179–2196. [CrossRef]
73. Akhtar, M.S.; Ghosal, D.; Ekbal, A.; Bhattacharyya, P.; Kurohashi, S. All-in-one: Emotion, sentiment and intensity prediction using a multi-task ensemble framework. *IEEE Trans. Affect. Comput.* **2022**, *13*, 285–297. [CrossRef]
74. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
75. Sharf, Z.; Rahman, S.U. Performing natural language processing on roman urdu datasets. *Int. J. Comput. Sci. Netw. Secur.* **2018**, *18*, 141–148.
76. Mehmood, K.; Essam, D.; Shafi, K.; Malik, M.K. Sentiment analysis for a resource poor language—Roman Urdu. *ACM Trans. Asian-Low-Resour. Lang. Inf. Process. (TALLIP)* **2019**, *19*, 1–15. [CrossRef]
77. Sermanet, P.; Eigen, D.; Zhang, X.; Mathieu, M.; Fergus, R.; LeCun, Y. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv* **2013**, arXiv:1312.6229.
78. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. In Proceedings of the Advances in Neural Information Processing Systems, Carson City, NV, USA, 5–10 December 2013; pp. 3111–3119.
79. Pennington, J.; Socher, R.; Manning, C. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 26–28 October 2014; pp. 1532–1543.
80. Bojanowski, P.; Grave, E.; Joulin, A.; Mikolov, T. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguistics* **2016**, *5*, 135–146. [CrossRef]
81. Sharf, Z.; Rahman, S.U. Lexical normalization of roman urdu text. *Int. J. Comput. Sci. Netw. Secur.* **2017**, *17*, 213–221.
82. Fang, X.; Zhan, J. Sentiment analysis using product review data. *J. Big Data* **2015**, *2*, 1–14. [CrossRef]
83. Shah, K.; Patel, H.; Sanghvi, D.; Shah, M. A comparative analysis of logistic regression, random forest and KNN models for the text classification. *Augment. Hum. Res.* **2020**, *5*, 1–16. [CrossRef]
84. Domingos, P.; Pazzani, M. On the optimality of the simple bayesian classifier under zero-one loss. *Mach. Learn.* **1997**, *29*, 103–130. [CrossRef]