Deep learning for conflicting statements detection in text

Vijay Lingam [1] , **Simran Bhuria** [1] , **Mayukh Nair** [1] , **Divij Gurpreetsingh** [1] , **Anjali Goyal** [1] , **Ashish Sureka** [Corresp. 1]

[1] Computer Science, Ashoka University, Sonepat, Haryana, India

Corresponding Author: Ashish Sureka
Email address: ashish.sureka@ashoka.edu.in

Background. Automatic contradiction detection or conflicting statements detection in text consists of identifying discrepancy, inconsistency and defiance in text and has several real world applications in questions and answering systems, multi-document summarization, dispute detection and finder in news, and detection of contradictions in opinions and sentiments on social media. Automatic contradiction detection is a technically challenging natural language processing problem. Contradiction detection between sources of text or two sentence pairs can be framed as a classification problem.

Methods. We propose an approach for detecting three different types of contradiction: negation, antonyms and numeric mismatch. We derive several linguistic features from text and use it in a classification framework for detecting contradictions. The novelty of our approach in context to existing work is in the application of artificial neural networks and deep learning. Our approach uses techniques such as Long short-term memory (LSTM) and Global Vectors for Word Representation (GloVe). We conduct a series of experiments on three publicly available dataset on contradiction detection: Stanford dataset, SemEval dataset and PHEME dataset. In addition to existing dataset, we also create more dataset and make it publicly available. We measure the performance of our proposed approach using confusion and error matrix and accuracy.

Results. There are three feature combinations on our dataset: manual features, LSTM based features and combination of manual and LSTM features. The accuracy of our classifier based on both LSTM and manual features for the SemEval dataset is 91.2%. The classifier was able to correctly classify 3204 out of 3513 instances. The accuracy of our classifier based on both LSTM and manual features for the Stanford dataset is 71.9%. The classifier was able to correctly classify 855 out of 1189 instances. The accuracy for the PHEME dataset is the highest across all datasets. The accuracy for the contradiction class is 96.85%.

Discussion. Experimental analysis demonstrate encouraging results proving our hypothesis that deep learning along with LSTM based features can be used for identifying contradictions in text. Our results shows accuracy improvement over manual features after applying LSTM based features. The accuracy results varies across datasets and we observe different accuracy across multiple types of contradictions. Feature analysis shows that the discriminatory power of the five feature varies.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26589v1 | CC BY 4.0 Open Access | rec: 1 Mar 2018, publ: 1 Mar 2018

Deep Learning for Conflicting Statements Detection in Text

Vijay Lingam[1], **Simran Bhuria**[1], **Mayukh Nair**[1], **Divij Gurpreetsingh**[1], **Anjali Goyal**[1], **and Ashish Sureka**[1]

[1]**Ashoka University, Haryana, India**

Corresponding author:
Ashish Sureka[1]

Email address: ashish.sureka@ashoka.edu.in

ABSTRACT

Background. Automatic contradiction detection or conflicting statements detection in text consists of identifying discrepancy, inconsistency and defiance in text and has several real world applications in questions and answering systems, multi-document summarization, dispute detection and finder in news, and detection of contradictions in opinions and sentiments on social media. Automatic contradiction detection is a technically challenging natural language processing problem. Contradiction detection between sources of text or two sentence pairs can be framed as a classification problem.

Methods. We propose an approach for detecting three different types of contradiction: negation, antonyms and numeric mismatch. We derive several linguistic features from text and use it in a classification framework for detecting contradictions. The novelty of our approach in context to existing work is in the application of artificial neural networks and deep learning. Our approach uses techniques such as Long short-term memory (LSTM) and Global Vectors for Word Representation (GloVe). We conduct a series of experiments on three publicly available dataset on contradiction detection: Stanford dataset, SemEval dataset and PHEME dataset. In addition to existing dataset, we also create more dataset and make it publicly available. We measure the performance of our proposed approach using confusion and error matrix and accuracy.

Results. There are three feature combinations on our dataset: manual features, LSTM based features and combination of manual and LSTM features. The accuracy of our classifier based on both LSTM and manual features for the SemEval dataset is 91.2%. The classifier was able to correctly classify 3204 out of 3513 instances. The accuracy of our classifier based on both LSTM and manual features for the Stanford dataset is 71.9%. The classifier was able to correctly classify 855 out of 1189 instances. The accuracy for the PHEME dataset is the highest across all datasets. The accuracy for the contradiction class is 96.85%.

Discussion. Experimental analysis demonstrate encouraging results proving our hypothesis that deep learning along with LSTM based features can be used for identifying contradictions in text. Our results shows accuracy improvement over manual features after applying LSTM based features. The accuracy results varies across datasets and we observe different accuracies across multiple types of contradictions. Feature analysis shows that the discriminatory power of the five feature varies.

1 INTRODUCTION

1.1 Research Motivation and Aim

Automatic contradiction detection or conflicting statements detection in text consists of identifying discrepancy, inconsistency and defiance in text (De Marneffe et al., 2008)(Lendvai et al., 2016)(de Marneffe et al., 2011)(Ritter et al., 2008). For example negation in a political debate by candidates taking a different position: one of the candidates says “I support the new anti-corruption law” and another candidates says that “I do not support the new anti-corruption law”. Another example of a contradictory pair of statements consisting of a numeric mismatch is: “More than 50 people died in the plane crash” and “10 people died in the plane crash”. These are relatively simple and straightforward examples of conflicting statements but the statements can be much more complex requiring deeper understanding, comprehension and inference of the text. For example a statement pair containing antonym is more complex than a simple negation: "I support the new anti-corruption law" and "I oppose the new anti-corruption law". Table 1 shows examples of three different types of contradiction statements considered in our experiments. The three different types of contradiction statements addressed in our work are: negation, antonyms and numeric mismatch.

There are several real world applications of contradiction detection and hence solutions for automatic contradiction detection has attracted the attention of several researchers in the field of machine learning, natural language processing and information retrieval. Harabagiu et al. motivate their work on contradiction detection by giving examples of applications such as question and answering and multi-document summarization systems which makes use of contradiction detection as one of the text processing step (Harabagiu et al., 2006). For example, if there are contradictory answers to a question in a question and answering system then a contradiction detection application can help in identifying such cases for intervention from the users requiring resolution of the contradiction between two answers (Harabagiu et al., 2006). Ennals et al. motivate the use of contradiction in text through an application called as dispute finder which is a web browser extension used for alerting the user in-case the user comes across the text which is disputed by a trusted sources (Ennals et al., 2010). Another interesting and useful application of contradiction detection in text is proposed by Tsytsarau et al. which consists of analysing user opinions posted on the web (Tsytsarau et al., 2011)(Tsytsarau et al., 2010). Tsytsarau et al. present an application of capturing diversity of sentiments on different topics expressed by users on the web (Tsytsarau et al., 2011)(Tsytsarau et al., 2010).

Contradiction detection is a technically challenging problem and a hard natural language processing task. Contradiction detection between sources of text or two sentence pairs can be framed as a classification problem. The most common approach for contradiction detection in text is to derive linguistic based features from text and then train or learn a classifier from hand-annotated examples to perform the categorization task. Contradiction detection in text is still not a fully solved problems and there are several limitations and research gaps in existing work (De Marneffe et al., 2008)(Lendvai et al., 2016)(de Marneffe et al., 2011)(Ritter et al., 2008). Our motivation is to build a solution for contradiction detection in text using a machine learning framework (particularly neural network) based on deriving linguistic evidences and textual features from text. Deep learning and deep artificial neural networks have become very popular in recent years due to their effectiveness in solving several pattern recognition and machine learning problems (Schmidhuber, 2015)(LeCun et al., 2015). Application of artificial neural networks and deep learning is a relatively unexplored and untapped area for the problem of contradiction detection in text. Our objective is to investigate the application of deep learning and artificial neural network for contradiction detection in text. Similarly techniques and methods like GloVe (Global vectors for word representation) (Pennington et al., 2014) and LSTM (long short-term memory networks) (Palangi et al., 2016) have gained lot of importance in the natural language processing and machine learning literature. Application of these techniques are unexplored for the contradiction detection and conflicting statement detection problem. Our motivation is to examine the application of GloVe and LSTM for feature extraction from sentences and for sentence representation. Specifically, our objective is to explore deep artificial neural network, GloVe and LSTM for solving the problem of contradiction detection in text. Our research aim is to conduct a series of experiments on several publicly available dataset to investigate the effectiveness of our proposed approach.

1.2 Related Work

Marie-Catherine De Marneffe et al. describe an approach for contradiction detection in text and also create a dataset for contradiction detection (De Marneffe et al., 2008). Their approach consists of creating a typed dependency graph produced by the Stanford parser followed by the step of alignment between text and hypothesis graphs. Their final step in the process consists of extracting contradiction features and applying logistic regression models for classifying whether a sentence pair is a contradiction or not (De Marneffe et al., 2008). Lendvai et al. create a Recognizing Textual Entailment (RTE) dataset based on naturally occurring contradiction in tweets posted during crisis events on the Twitter micro-blogging platform (Lendvai et al., 2016). They created the dataset which enables researchers in the area of natural language processing and information retrieval to build statistical models for drawing on semantic inferences across microblog posts and text (De Marneffe et al., 2008). Harabagiu et al. describe a framework for identifying presence of contradictions between a pair of text such as contradictions occurring due to presence of

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26589v1 | CC BY 4.0 Open Access | rec: 1 Mar 2018, publ: 1 Mar 2018

	Type	Sentence 1	Sentence 2
1	Negation	I keep thinking about you	I never think about you
2	Negation	It concerns my brother	It does not concern my brother
3	Negation	Nobody is on a motorcycle and is standing on the seat	Someone is on a black and white motorcycle and is standing on the seat
4	Antonym	I can't confidently tell you yet	I can't diffidently tell you yet
5	Antonym	I've been thinking about Tom a lot	I've been thinking about Tom a little
6	Antonym	Why don't you let me go	Why do you let me go
7	Numeric Mismatch	Jennifer Hawkins is the 21-year-old beauty queen from Australia	Jennifer Hawkins is Australia's 20-year-old beauty queen
8	Numeric Mismatch	Four people were killed and a US helicopter shot down in Najaf	Five people were killed and an American helicopter was shot down in Najaf
9	Numeric Mismatch	Eight million Americans have hyperhidrosis	A recent study estimated that 12 million Americans have hyperhidrosis

Table 1. Examples showing different types of conflicting statements

negation and antonyms (Harabagiu et al., 2006). Their proposed approach consists of several modules such as linguistic pre-processing, lexical alignment, feature extraction and classification (Harabagiu et al., 2006). They evaluate their system on multiple datasets. For example, they evaluate their contrast detection system using a text corpus consisting of 10000 instances of discourse relations extracted from publicly available newswire documents (Harabagiu et al., 2006).

Rob Ennals et al. describe a tool called as Dispute Finder which is deployed as a web browser extension and alerts the reader then the information being read by the reader online is disputed by a trusted source (Ennals et al., 2010). Their approach is based on building a database or repository of disputed claims by crawling various websites on the Internet and maintaining a list of disputed claims (Ennals et al., 2010). Their approach is based on invoking a textual entailment procedure inside the web browser extension (Ennals et al., 2010). Mikalai Tsytsarau et al. present a method for finding sentiment based contradictions in text (Tsytsarau et al., 2011). Their focus is on analysis of user opinions expresses on the Web such as on social media websites and blogosphere (Tsytsarau et al., 2011). They develop a method of measuring contradictions based on the mean value and variance of sentiments among different texts (Tsytsarau et al., 2011). Alan Ritter et al. present a case-study on contradiction detection using functional relations (Ritter et al., 2008). Their proposed algorithm is domain dependent which automatically discovers phrases denoting functions with a good precision (Ritter et al., 2008). They investigate the effectiveness of their approach based on harvesting sentence pairs from the Web that appear contradictory (Ritter et al., 2008). Shih et al. focus on the problem of the lack of background knowledge for contradiction detection systems (Shih et al., 2012) . Their approach is based on measuring the availability of mismatch conjunction phrases (MCP) and they demonstrate the effectiveness of their approach by conducting experiments on three different configurations (Shih et al., 2012). Daisuke Kawahara et al. present a system which displays contradictory and contrastive relations among statements expresses on a particular topic on selected web pages (Kawahara et al., 2010). Their approach works in an unsupervised manner in which cross-document implicit contrastive relations between statements are extracted (Kawahara et al., 2010).

1.3 Research Contributions

In context to existing work, the study presented in this paper makes the following novel and unique research contributions:

Novel Approach based on Deep Learning To the best of our knowledge and based on our analysis of the existing literature on contradiction detection, our proposed approach is the first study using techniques such as deep learning, Long short-term memory (LSTM) and Global Vectors for Word Representation (GloVe). The features used for contradiction detection and the overall solution

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26589v1 | CC BY 4.0 Open Access | rec: 1 Mar 2018, publ: 1 Mar 2018

architecture is novel.

Experimental Evaluation on Diverse Dataset We demonstrate the effectiveness of our approach by conducting experiments on multiple diverse dataset. The research community on contradiction detection in text has been contributing dataset on contradiction detection and one of the uniqueness of our presented in this paper is an in-depth experimental evaluation on multiple dataset and not just one or two corpus.

Dataset Contribution In addition to conducting experiments on multiple existing dataset, we also create more dataset and manually annotate every sentence paper. We make our dataset publicly available on Figshare (Lingam et al., 2018).

2 MATERIALS AND METHOD

2.1 Solution Approach and Research Framework

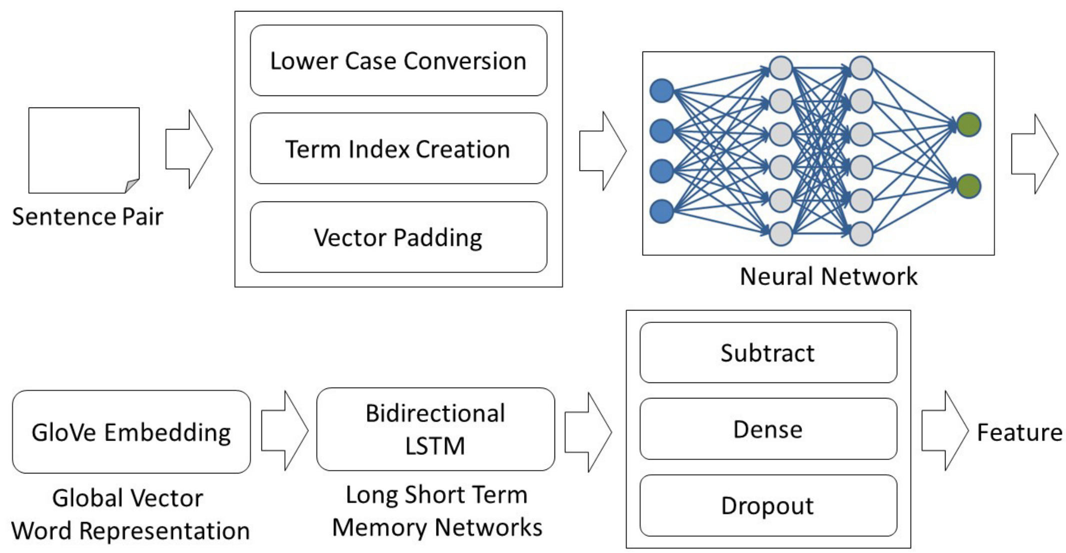

Figure 1. High Level Solution Approach and Research Framework Diagram - LSTM Based Feature Extraction

Figures 1 and 2 shows our proposed solution approach. As shown in Figures 1 and 2, our approach consists of multiple steps. The first step in the process is to convert all the input text in lower case. We do not perform any stop word removal or term stemming. Terms like "and" and "not are useful features for contradiction detection (such as negation). Similarly, we do not remove any numeric values as numbers such as "10" or "100" are useful for contradiction detection (such as numeric mismatch). We have written all our programs in the Python programming language and hence we use the TensorFlow Python library for conducting experiments on deep learning and neural networks. We use the TensorFlow machine learning system for training and testing a predictive model for contradiction detection in text (Abadi et al., 2016a)(Abadi et al., 2016b). We use TensorFlow for all our experimentations presented in this paper as TensorFlow provides a wide variety of functionalities and is quite flexible to support research and experimentation. Another justification behind our usage of TensorFlow is that it is an open source project which has a large community of users and developers around it.

We combine the training and test instances for a particular dataset and create a corpus. We then compute all the unique terms in the corpus. Each term in the corpus is given an index id. We convert every sentence in our dataset into a vector containing the index id of the word present in the sentence. For example, if the sentence is "apple on the table" then it gets converted into a vector [12 30 7 44] in which the index of the terms (in the vocabulary for the corpus) "apple", "on", "the" and "table" are 12, 30, 7 and

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26589v1 | CC BY 4.0 Open Access | rec: 1 Mar 2018, publ: 1 Mar 2018

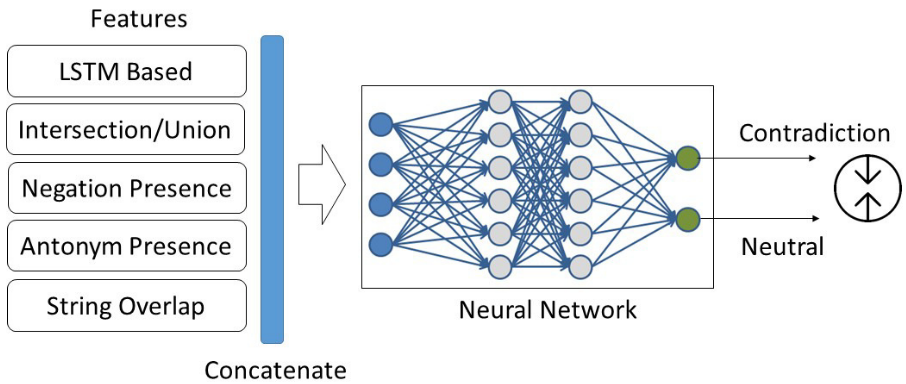

Figure 2. High Level Solution Approach and Research Framework Diagram - Contradiction and Neutral Detection in Text using Neural Network and Five Features

44 respectively. As illustrated in Figure 1, we perform an operation called as padding with a dimension of 40. The size of each vector is made 40 by inserting 0 in empty elements of the vector. For example, if the length of sentence "A" is 7 and length of sentence "B" is "15" then 33 0's are inserted in the vector representation for sentence "A" and 25 0's are inserted in the vector representation for sentence "B". We perform an operation called as GloVe[1] embedding in which each term in our sentence is converted into a vector (word to vector) which is then used as a feature for the natural language processing task of contradiction detection in text (Pennington et al., 2014). GloVe embedding helps in creating a word to vector representation which captures linguistic regularities and helps in performing vector operations such as subtract (as shown in the Figure 1) and addition. As shown in Figure 1, we can perform operations such as vector("research") – "vector("journal") on the real-valued vector obtained as a result of GloVe embedding. We use an embedding dimension of 300 while creating GloVe embedding. Each word in our input is represented as a real-valued vector with a dimension of 300. As shown in Figure 1 the vector is used for creating one of the features for the text classification task of contradiction or neutral detection.

We apply a bidirectional LSTM (Long Short Term Memory networks) approach which is an extension of the traditional LSTM. We use the bidirectional LSTM deep neural networks as they have shown encouraging results on a variety of domains and dataset. RNN (Recurrent Neural Networks) with LSTM is a well-known technique for the purpose of encoding an English sentence into a vector such the semantic meaning of the sentence is contained in the vector (Hochreiter and Schmidhuber, 1997)(Palangi et al., 2016). We apply an RNN with LSTM based approach as learning a good representation of the sentence pair which needs to be classified is important for the task of contradiction or neutral sentences detection. In an attempt to improve the accuracy of our system, we engineered a few features. We noticed a significant increase in accuracy upon integrating these features with the features generated by the neural network. As shown in Figure 2, we create the following four features and implement them in our system:

Jaccard Coefficient Jaccard Coefficient (also known as Intersection over Union - IOU) is a widely used metric in information retrieval applications used to measure similarity between two text. In our case, it is simply a fraction with the number of words common to both sentences as the numerator and the number of total words in both sentences as the denominator. The coefficient captured the relation between the amount of similarity between the two sentences and the existence of a contradiction between them. Computing similarity is useful in sentence pair on the same topic and using similar vocabulary.

Negation It is a binary feature that takes the values true or false. It is true when one of the sentences in the given sentence pair contains one of these words no, never, not , nothing, no one, without, nobody and the other does not contains any words from our predefined negation list. The idea

[1] `https://nlp.stanford.edu/projects/glove/`

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26589v1 | CC BY 4.0 Open Access | rec: 1 Mar 2018, publ: 1 Mar 2018

	Sentence Type	SEMEVAL	Stanford	Pheme
Training Instances	Neutral	2536	779	606
	Contradiction	1286	294	300
Testing Instances	Neutral	2793	865	260
	Contradiction	720	325	127

Table 2. Experimental Dataset : 3 different dataset, training and testing instances and two classes (neutral and contradiction)

here was to capture contradictions where one sentence expresses a negative sentiment while the other one does not. Clearly, this feature alone cannot discriminate between a contradictory and a non-contradictory statement. However, the feature can be useful while analysing sentence pairs (short sentences) on the same topic and using similar vocabulary.

IsAntonym This feature is very intuitive and self-explanatory. It takes the value 0 if none of the words present in one of the sentences have their antonyms in the other sentence. It takes the value 1 otherwise. We check the words from each of the sentences against a set of antonyms that we assembled from The Non-Official Characterization (NOC) after adding 47 antonyms from our end (Veale, 2016). The final set contains 3714 antonyms. If a word from any of the sentences is found on our antonym list, we fetch its antonym from the set and check whether that word is present in the other sentence. If it is present, then the value is 1, otherwise it is 0. The list is specific to our dataset and can be enhanced as more diverse dataset is added.

Overlap Coefficient The Overlap Coefficient is another similarity metric like Jaccard Coefficient. It measures the overlap between two sets and is computed as the size of the intersection divided by the smallest size of the two sets. Overlap coefficient captures the similarity well when the difference between the sizes of the two sentences is large.

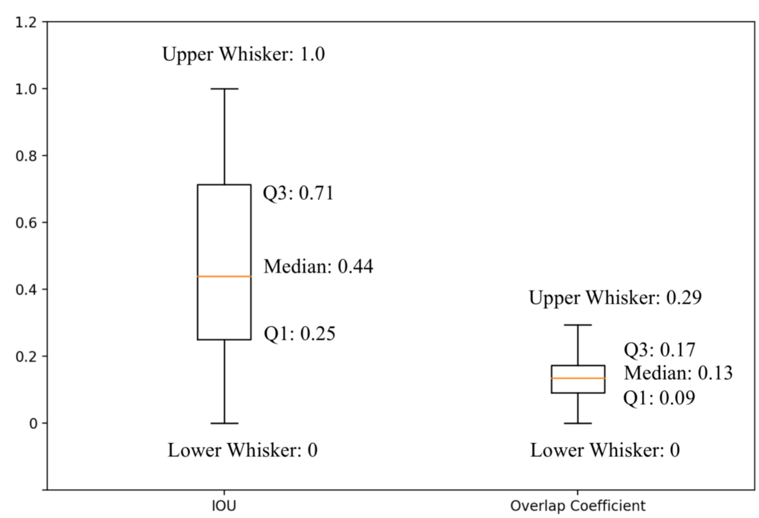

Figure 3. Boxplot for IOU (Intersection over Union) and Overlap Coefficient Feature Values

2.2 Experimental Dataset

Table 2 shows the experimental dataset details. We conduct experiments on three different dataset to increase the generalizability of our results and conclusions. All the dataset is publicly available and hence our results can be used for benchmarking and comparison. One of the three datasets (SemEval) is downloaded from SemEval-2014[2] which was an international workshop on semantic evaluation conducted in Dublin (Ireland). Semantic Evaluation referred to as SemEval is a well-known workshop organized by the Special Interest Group on the Lexicon of the Association for Computational Linguistics[3] (ACL). The SemEval dataset originally consisted of 665 contradiction sentence pairs as part of the training instances. We manually created 621 contradiction sentence pairs to increase the count to 1286 so that the machine learning classification algorithm has enough number of contradiction sentence pairs for training and model building. As shown in Table 2, the number of contradiction class instances and neutral class instances in the test dataset are 720 and 2793 respectively. We experiment with the same dataset (Stanford) as used by Marneffe et al. (De Marneffe et al., 2008) for their work on finding contradictions in text. The Stanford Contradiction Corpora[4] used in our experiments can be downloaded from the Stanford Natural Language Processing Group website. As shown in Table 2, the dataset is not balanced and the number of instances of contradiction class is less than the number of instances belonging to the neutral class. The dataset is of high quality as it has been annotated by the authors of the paper by Marneffe et al. (De Marneffe et al., 2008) as well as various students and faculty at Stanford. The number of contradiction sentence pairs in the training and testing instanced for the Stanford dataset are 294 and 325 respectively. Another dataset that we use is the PHEME RTE (Recognizing Textual Entailment) dataset[5]. The dataset is created and used by Lendvai et al. (Lendvai et al., 2016) for their work on detecting contradiction and entailment. The PHEME dataset is also imbalanced with respect to the contradiction class. As shown in Table 2, the number contradiction sentence pairs in the training and testing instanced for the Stanford dataset are 299 and 127 respectively. The PHEME dataset (Chebedo dataset) is diverse and different in comparison to SemEval and Standford dataset as the PHEME dataset is based on naturally occurring contradictions on Tweets posted on Twitter related to crisis events (Lendvai et al., 2016).

SemEval

	LSTM		Manual Features		LSTM+Manual Features	
	CNT	NOT	CNT	NOT	CNT	NOT
CNT	560	160	617	103	599	121
NOT	148	2645	208	2585	188	2605

Stanford

	LSTM		Manual Features		LSTM+Manual Features	
	CNT	NOT	CNT	NOT	CNT	NOT
CNT	23	302	0	325	9	316
NOT	68	796	0	864	18	846

PHEME

	LSTM		Manual Features		LSTM+Manual Features	
	CNT	NOT	CNT	NOT	CNT	NOT
CNT	119	8	3	124	123	4
NOT	9	251	0	260	0	260

Table 3. Confusion or Error Matrix

3 RESULTS

3.1 Box Plot for Feature Values

There are several features or independent variables for our classification problem on contradiction detection in text. The range and scale of all the independent variables or predictors are not same as the formula and processing for computing the feature value is dependent on the type of the feature. We apply techniques to standardize the range of our independent variables. Data normalization and scaling is an important

[2] http://alt.qcri.org/semeval2014/
[3] https://www.aclweb.org/portal/
[4] https://nlp.stanford.edu/projects/contradiction/
[5] https://www.pheme.eu/2016/04/12/pheme-rte-dataset/

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26589v1 | CC BY 4.0 Open Access | rec: 1 Mar 2018, publ: 1 Mar 2018

data pre-processing step and is done before applying the classification algorithms in a machine learning processing pipeline (Graf et al., 2003). We rescale the range of all our features to a scale in the range of 0 to 1. Figure 3 shows a boxplot for IOU (Intersection over Union) and Overlap Coefficient feature values. We display the box plot of two the features as an illustration. Figure 3 shows the descriptive statistics using a boxplot visualization presenting the summary of the two features in-terms of the central tendency, dispersion and spread. Figure 3 reveals that the median value for the IOU feature is 0.44 and the median value for the overlap coefficient feature is 0.13. We observe that the feature values are different for different instances and hence has a potential for discriminating the instances into classes.

We study the median values of all the features in our feature-set as the median value is the measure of the centrality and can provide us with useful insights on the skewness of the data. The boxplot in Figure 3 displays the first and third quartile values ($Q1$ and $Q3$) for IOU and overlap coefficient feature. The $Q1$ and $Q3$ are used by us to compute the interquartile range indicating the variability around the mean and understanding factors influencing the discriminatory power of the feature. From the numerical summary presented in the boxplot of Figure 3, we infer that the values for the two features are scattered and have a spread. The IOU and overlap coefficient feature values are diverse and contains several values between the largest (= 1) and the smallest (= 0). The spread and descriptive statistics for the features are different and we observe that they are not correlated and provide different perspectives.

3.2 Confusion Matrix (for all the three dataset: SemEval, Stanford and PHEME)

Table 3 shows the confusion or error matrix describing the performance of deep artificial neutral network while considering 3 different feature sets: LSTM based features, manual features and combination of LSTM based and manual features. In the study presented in this paper, we use confusion matrices and accuracy measure for our statistical model evaluation. A confusion matrix is a way to precisely and in a tabulated form represent prediction results obtained from a machine learning classifier (Manning et al., 2008). The confusion matrix represents the number of correctly classified instances in the test dataset and also incorrectly classified instances in the test dataset by a machine learning algorithm (Manning et al., 2008). The rows of the confusion matrix (refer to Table 3) lists all the predicted classes and the columns of the confusion matrix lists all the actual classes. The diagonal elements in a confusion matrix represents the total number (or percentage) of correctly classified instances, i.e. the number of instances which were corrected predicted to the actual class by the learning algorithm. The elements other than diagonal elements in the confusion matrix represents the number of incorrectly classified (misclassification) instances.

There are 3 different datasets used in this work for experimental evaluation: SemEval, Stanford and RTE. Hence, we have a total of 9 confusion matrices. There are 3 confusion matrices for each project: 1 confusion matrix showing the performance when LSTM based features are used for prediction, 1 confusion matrix using manual features for prediction and 1 confusion matrix using the combination of LSTM and manual features for prediction. There are two classes in our dataset: CNT and NOT. The rows of the confusion matrix represent the actual class and the columns represent the predicted class. We present the results in the form of confusion matrix as our objective in this work was to study both classifications, CNT as well as misclassifications, NOT. Table 3 reports the false positives, false negatives, true positives, and true negatives for each feature set. The confusion matrices are for the test data of each project. Table 3 reveals that for SemEval dataset, LSTM feature based prediction can correctly classify 560 CNT instances. This is termed as true positives. True positives are cases which are correctly classified by the learning algorithm. For example, out of 720 contradiction sentence pairs in test set, 560 sentence pairs were correctly classified as CNT when LSTM based features are used. True negatives are cases which were not CNT and were not classified as CNT. For example, out of 2793 sentence pairs in test set, 2645 sentence pairs were correctly classified as NOT when manual features are used. Tables 3 also reveals the number of test cases where the learning algorithm is predicting wrong label. For example, 160 sentence pairs belonging to CNT class were misclassified by learning algorithm and were predicted as NOT. This is known as False negative. 148 sentence pairs belonging to NOT class were misclassified as CNT. This is known as False Positive. Similarly, when using manual features for prediction 617 sentence pairs were correctly classified as CNT and 2585 sentence pairs were correctly classified as NOT by learning algorithm. However, 103 sentence pairs of CNT class and 208 sentence pairs of NOT were misclassified. Using combination of LSTM and manual features, learning algorithm correctly predicted 3204 sentence pairs were correctly predicted. 599 sentence pairs of CNT class and

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26589v1 | CC BY 4.0 Open Access | rec: 1 Mar 2018, publ: 1 Mar 2018

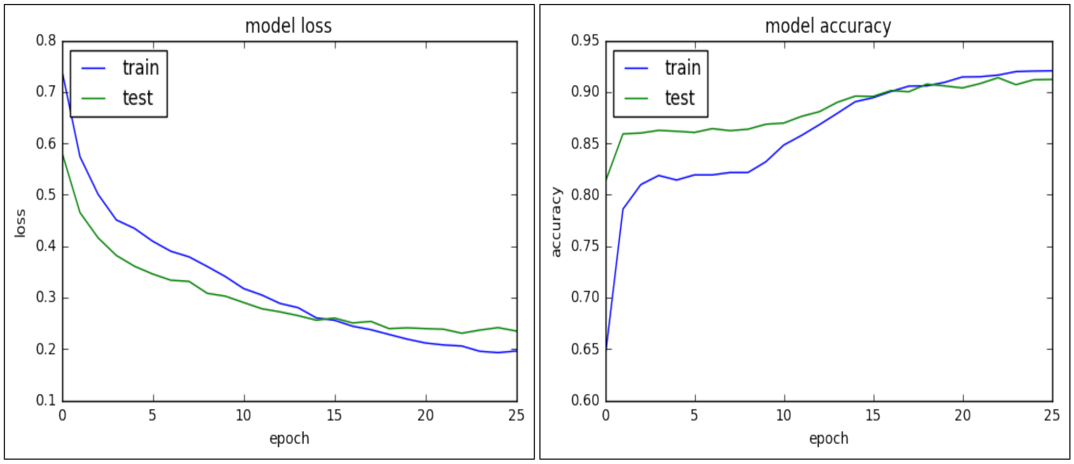

Figure 4. Neural network model loss and model accuracy.

2605 sentence pairs of NOT are correctly predicted by deep artificial neural network. However, the learning algorithm misclassified 121 CNT sentence pairs to NOT class and 188 NOT sentences pairs to CNT class.

Similarly, for Stanford project, there were a total of 1190 sentence pairs in test set. Out of these 1190 sentence pairs, 325 belongs to CNT class and 865 sentence pairs belong to NOT class (ground truth). Table 3 depicts that using LSTM based features, 819 sentence pairs were correctly classified and 370 sentence pairs were misclassified. 23 sentence pairs of CNT class and 796 sentence pairs of NOT class were correctly predicted whereas 68 sentence pairs belonging to NOT class were misclassified as CNT and 302 sentence pairs belonging to CNT class were misclassified as NOT by learning algorithm. Using manual features, the learning algorithm classified all 1190 sentence pairs to NOT class. Hence, all the contradiction pairs got misclassified by the classifier. Using combination of LSTM and manual features, 854 sentence pairs were classified correctly by machine classifier whereas 334 sentence pairs were misclassified. 9 sentence pairs of CNT class and 846 sentence pairs of NOT class were correctly predicted whereas 18 sentence pairs belonging to NOT class were misclassified as CNT and 316 sentence pairs belonging to CNT class were misclassified as NOT by the machine learning algorithm.

For Pheme dataset, there were a total of 387 sentence pairs in test set. Out of these, 127 sentence pairs belong to CNT class and 260 sentence pairs belong to NOT class (ground truth). Table 3 depicts that using LSTM based features, 370 sentence pairs were correctly classified and 17 sentence pairs were misclassified. 119 sentence pairs of CNT class and 251 sentence pairs belonging to NOT class were correctly predicted by the machine learning algorithm. 9 sentence pairs belonging to NOT class were misclassified as CNT and 8 sentence pairs belonging to CNT class were misclassified as NOT by learning algorithm. Using manual features, the learning algorithm classified 384 sentence pairs to NOT class. However, 124 sentence pairs predicted as NOT belongs to CNT class and got misclassified. 3 sentence pairs were correctly classified as CNT and 260 sentence pairs were correctly classified as NOT. Using combination of LSTM and manual features, 383 sentence pairs were classified correctly by machine classifier whereas only 4 sentence pairs were misclassified. 123 sentence pairs of CNT class and 260 sentence pairs of NOT class were correctly predicted whereas only 4 sentence pairs belonging to CNT class were misclassified as NOT by the machine learning algorithm.

3.3 Neural network model loss and accuracy

Figure 4 presents about our deep learning model and neural network model (using LSTM and manual features setting for the SemEval dataset) performance over time during training and testing. Our objective was to visualize the performance of our deep learning models. We do it using Keras[6] which is a high-level neural networks API implemented in Python and capable of running on top of TensorFlow. We used Keras as it is in Python and compatible with rest of our code. Also, Keras allows us to do fast prototyping and experimentation. Figure 4 consists of two graphs hosing the training metrics for each epoch. We present

[6] `https://keras.io/`

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26589v1 | CC BY 4.0 Open Access | rec: 1 Mar 2018, publ: 1 Mar 2018

SemEval

	LSTM	Manual Features	LSTM+Manual Features
CNT	77.78%	85.69%	83.19%
NOT	94.70%	92.55%	93.27%

Stanford

	LSTM	Manual Features	LSTM+Manual Features
CNT	7.08%	0%	2.77%
NOT	92.13%	100%	97.92%

Pheme

	LSTM	Manual Features	LSTM+Manual Features
CNT	93.70%	2.36%	96.85%
NOT	96.54%	100%	100%

Table 4. Accuracy Results of NOT and CNT class for 3 datasets across 3 feature sets

the graph for the loss as well as the accuracy for our classification problem of contradiction detection in sentence pairs. Figure 4 reveals how our deep learning model converges and also presents insights on the speed of convergence over epochs. From the accuracy plot, we observe that the model gets trained until the trend for accuracy starts becoming flat and does not rise. The loss graph in Figure 4 shows that the model has a comparable performance on the training and test dataset. We study and create the accuracy and loss graphs for our deep learning models for all the dataset and for the feature combinations and present one such result as an illustration in Figure 4.

3.4 Classification Accuracy

Table 4 shows the detailed performance results of deep artificial neural network while using different feature sets. Classification accuracy is computed by summing the value of true positives and true negatives and dividing it by the total number of instances in the test dataset. Table 4 presents the accuracy results for both classes CNT as well as NOT. Table 4 reveals that for SemEval dataset, LSTM feature based prediction achieves an accuracy of 77.78% for CNT class and 94.70% for NOT class. Similarly, using manual features, the classifier achieves an accuracy of 85.69% for CNT class and 92.55% for NOT class. Using combination of LSTM and manual features, classifier achieves an accuracy of 83.19% for CNT class and 93.27% for NOT class. We observe that among the 3 feature sets, Manual feature set is best capable to predict CNT class whereas using combination of LSTM and manual features is more useful while predicting NOT class.

Similarly, for Stanford dataset, LSTM based feature set achieves accuracy of 7.08% for CNT class and 92.13% for NOT class. While using manual features for prediction, all sentence pairs in test set were classified as NOT class. Hence, using manual features, the accuracy for CNT class is 0% as all sentence pairs of CNT got misclassified as NOT whereas the accuracy for NOT class is 100%. Using combination of LSTM and manual features, the learning model achieves accuracy value of 2.77% for CNT class ad 97.92% for NOT class. We observe that among 3 feature sets, LSTM based feature set achieved highest performance for CNT class whereas using combination of LSTM and manual features is useful for prediction of NOT class. Although, manual feature set achieved an accuracy of 100% for NOT class but this is due to the fact that classifier is predicting all instances as NOT. Hence, manual feature based classification for Stanford dataset is not a good measure. For Pheme dataset, LSTM based feature set achieves an accuracy of 93.70% for CNT class and 96.54% for NOT class. While using manual features for prediction, the accuracy achieved for CNT class is 2.36% and NOT class is 100%. Using combination of LSTM and manual features, the learning model achieves accuracy value of 96.85% for CNT class and 100% for NOT class. We notice that among the 3 feature sets, using combination of LSTM and manual feature is most useful for prediction of CNT class whereas using 2 feature sets: Manual and combination of LSTM and manual features, classifier can precisely predict NOT class with an accuracy of 100%.

Table 5 represents the frequency distribution of sentence pairs of contradiction class in test set among different types of contradictions. In this work, we consider 4 different types of contradictions: Antonyms, Numeric mismatch, Negation and Others. For SemEval dataset, there were a total of 720 instances of CON class in test set. Out of these 720 sentence pairs, 66 sentence pairs belong to antonym type of contradiction,

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26589v1 | CC BY 4.0 Open Access | rec: 1 Mar 2018, publ: 1 Mar 2018

Type of Contradiction	SemEval	Stanford	Pheme
Antonym	66	30	0
Numeric	0	47	124
Negation	632	40	0
Others	22	208	3
TOTAL	**720**	**325**	**127**

Table 5. Frequency Distribution of Sentence Pairs of Contradiction Type in Test Set of 3 Datasets among Different Type of Contradictions

Type of Contradiction	SemEval	Stanford	Pheme
Antonym	13.64%	0%	-
Numeric	-	2.13%	98.39%
Negation	91.61%	10%	-
Others	50%	1.92%	66.67%

Table 6. Accuracy Results of Deep Artificial Neural Network using LSTM + Manual Features for 3 Datasets Across 4 Different Contradiction Types

632 sentence pairs are negation of each other and 22 sentence pairs belongs to contradictions other than antonyms, numeric mismatch and negation. Similarly, for Stanford dataset, there were 325 sentence pairs in test set of CON type. Out of these 325 instances, 30 sentence pairs belong to antonym type of contradiction, 47 sentence pairs have numeric match, 40 sentence pairs are negation of each other and 208 sentence pairs belongs to contradictions other than antonyms, numeric mismatch and negation types. For Pheme dataset, the test set contains a total of 127 contradiction sentence pairs. From these 127 contradictory sentence pairs, 124 sentence pairs belong to contradiction of type numeric mismatch and 3 sentence pairs belong to contradictions of type other than antonyms, numeric mismatch and negation.

Table 6 shows the detailed performance result of deep artificial neural network while using LSTM + Manual Features in 3 different datasets across 4 different types of contradictions. For SemEval dataset, out of 66 sentence pairs of antonym type, 9 sentence pairs got correctly classified as contradiction resulting in an accuracy value of 13.64% for antonym class. Similarly, out of 632 negation type sentence pairs in test set, 579 sentence pairs were correctly classified as contradictions. This results in an accuracy value of 91.61% corresponding to negation type of contradiction. For others type, accuracy of 50% is achieved. This shows that among the different type of contradictions in SemEval dataset, sentence pairs with negation type of contradiction is detected most accurately by deep artificial neural network. Similarly, for Stanford dataset, none of the sentence pairs of antonym type got classified correctly leading to 0% classification accuracy. For contradictions containing numeric mismatch, 1 sentence pair out of 47 sentence pairs got classified correctly. This results in an accuracy of 2.13% for predicting numeric mismatch type contradiction. For negation and others types of contradictions, accuracy value is 10% and 1.92% respectively. For Pheme dataset, out of 124 contradiction pairs belonging to numeric type, 122 sentence pairs got classified correctly as contradictions resulting in accuracy value of 98.39%. For others type of contradiction, 3 out of 4 sentence pairs got correctly classified in CON class leading to an accuracy result of 66.67% for contradiction detection. We found that among the two types of contradiction sentence pairs (numeric mismatch and others) present in PHEME test set, numeric mismatch type of contradictory sentence pairs are most accurately classified into CON class.

4 DISCUSSION

We present our detailed experimental results and insights in the previous section. In this section, we do not discuss our results and insights and rather present our analysis on the threats to validity.

4.1 Threats to Validity

The work presented in this paper is a machine learning based empirical study consisting of an empirical evaluation. The hypothesis, claims and solution approaches presented in our paper is empirically assessed.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26589v1 | CC BY 4.0 Open Access | rec: 1 Mar 2018, publ: 1 Mar 2018

In this section, we discuss our views on how we went about maximizing internal and external validity related to our work. We present an analysis of some of the possible and inevitable threats to validity in our experiments. While we have tried our best to mitigate various types of threats to validity issues, as mentioned by Siegmund et al., there is always an inherent trade-off between internal and external validity (Siegmund et al., 2015). One threat to validity is the researcher bias. Researcher bias depends on who does the work and arises because of the researcher (Shepperd et al., 2014). The predictive performance of artificial neural network and machine learning classifiers can be influenced by several issues such as the choice of classification parameters by the researchers, dataset used by the researchers as well as reporting protocols (Shepperd et al., 2014). Another threat to validity is related to the changes in the independent variables (or features). Are the independent variables used in our experiments indeed responsible for the observed variation in the dependent or target variable (in our case whether a given sentence pair contains contradiction or not). In order to mitigate this specific threat to validity, we conducted experiments on multiple types of publicly available dataset and conducted feature analysis by computing its descriptive statistics and visualizing it using boxplots. We extract different types of features from the sentence pair but we do not perform any link or meta-data analysis which can be considered as extraneous variables or confounding variables that influencing the dependent variable (this is one possible threat to validity). To mitigate external validity on whether our results are applicable to other classes or sub-categories, we created conduct experiments and investigate the performance on three different types of contradictions. However, we believe that more experiments are required to investigate if the study results and approach is applicable to other types of contradictions not covered by us. The dataset that we contributed and made publicly available was annotated and verified by more than one person (authors of this paper) to ensure that the dataset annotation is of high quality and there are no annotation and measurement errors. We also executed the experiments more than once to ensure that there are no errors while conducting the experiments and that our results are replicable. While our results shows relationship between the dependent variable and independent variables, we believe more experiments on a large dataset and dataset belonging to more types of contradictions is needed to strengthen our conclusions showing that the variables accurately model our hypothesis.

5 CONCLUSION

We present a method based on deep learning, artificial neural networks, long short-term memory and global vectors for word representation for conflicting statements detection in text. Our objective is to build a system to identify inconsistencies and defiance in text. We frame the problem of contradiction detection in text as a classification problem which takes a sentence pair as inputs and outputs a binary value indicating whether the sentence pairs are contradictory. We first derive linguistic evidences and textual features from the sentence pair such as presence of negation, antonyms, intersection and string overlaps. We apply artificial neural network, long short-term memory based feature and GloVe embedding. We conduct experiments on three dataset for examining the generalizability of our proposed approach. We also manually annotate new dataset and contribute it to the research community by making it publicly available. There are three feature combinations on our dataset: manual features, LSTM based features and combination of manual and LSTM features. The accuracy of our classifier based on both LSTM and manual features for the SemEval dataset is 91.2%. The accuracy of our classifier based on both LSTM and manual features for the Stanford dataset is 71.9%. The classifier was able to correctly classify 855 out of 1189 instances. The accuracy for the PHEME dataset is the highest across all datasets. The accuracy for the contradiction class is 96.85%. Our classifier performed best on the PHEME dataset. The accuracy of the classifier for the contradiction class on SemEval dataset having both LSTM and manual features is 83.19%. The accuracy of the classifier for the numeric mismatch type of contradiction on the PHEME dataset is 98.39%. The IOU and overlap coefficient feature values are diverse and contains several values between the largest (= 1) and the smallest (= 0). The spread and descriptive statistics for the features are different and we observe that they are not correlated and provide different perspectives. Overall, our experimental analysis demonstrates that it is possible to accurately detect contradictions in short sentence pairs containing negation, antonym and numeric mismatch using deep learning techniques.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26589v1 | CC BY 4.0 Open Access | rec: 1 Mar 2018, publ: 1 Mar 2018

REFERENCES

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al. (2016a). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016b). Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283.

De Marneffe, M.-C., Rafferty, A. N., and Manning, C. D. (2008). Finding contradictions in text. In *ACL*, volume 8, pages 1039–1047.

de Marneffe, M.-C., Rafferty, A. R., and Manning, C. D. (2011). Identifying conflicting information in texts. *Handbook of Natural Language Processing and Machine Translation: DARPA Global Autonomous Language Exploitation*.

Ennals, R., Trushkowsky, B., and Agosta, J. M. (2010). Highlighting disputed claims on the web. In *Proceedings of the 19th international conference on World wide web*, pages 341–350. ACM.

Graf, A. B., Smola, A. J., and Borer, S. (2003). Classification in a normalized feature space using support vector machines. *IEEE Transactions on Neural Networks*, 14(3):597–605.

Harabagiu, S., Hickl, A., and Lacatusu, F. (2006). Negation, contrast and contradiction in text processing. In *AAAI*, volume 6, pages 755–762.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Kawahara, D., Inui, K., and Kurohashi, S. (2010). Identifying contradictory and contrastive relations between statements to outline web information on a given topic. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 534–542. Association for Computational Linguistics.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436.

Lendvai, P., Augenstein, I., Bontcheva, K., and Declerck, T. (2016). Monolingual social media datasets for detecting contradiction and entailment. In *LREC*.

Lingam, V., Bhuria, S., Nair, M., Gurpreetsingh, D., Goyal, A., and Sureka, A. (2018). Dataset for conflicting statements detection in text.

Manning, C. D., Raghavan, P., Schütze, H., et al. (2008). *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge.

Palangi, H., Deng, L., Shen, Y., Gao, J., He, X., Chen, J., Song, X., and Ward, R. (2016). Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 24(4):694–707.

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

Ritter, A., Downey, D., Soderland, S., and Etzioni, O. (2008). It's a contradiction — no, it's not: a case study using functional relations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 11–20. Association for Computational Linguistics.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks*, 61:85–117.

Shepperd, M., Bowes, D., and Hall, T. (2014). Researcher bias: The use of machine learning in software defect prediction. *IEEE Transactions on Software Engineering*, 40(6):603–616.

Shih, C., Lee, C., Tsai, R. T., and Hsu, W. (2012). Validating contradiction in texts using online co-mention pattern checking. *ACM Transactions on Asian Language Information Processing (TALIP)*, 11(4):17.

Siegmund, J., Siegmund, N., and Apel, S. (2015). Views on internal and external validity in empirical software engineering. In *Software Engineering (ICSE), 2015 IEEE/ACM 37th IEEE International Conference on*, volume 1, pages 9–19. IEEE.

Tsytsarau, M., Palpanas, T., and Denecke, K. (2010). Scalable discovery of contradictions on the web. In *Proceedings of the 19th international conference on World wide web*, pages 1195–1196. ACM.

Tsytsarau, M., Palpanas, T., and Denecke, K. (2011). Scalable detection of sentiment-based contradictions. *DiversiWeb, WWW*, 1:9–16.

Veale, T. (2016). Round up the usual suspects: Knowledge-based metaphor generation. In *Proceedings of the Fourth Workshop on Metaphor in NLP*, pages 34–41.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.26589v1 | CC BY 4.0 Open Access | rec: 1 Mar 2018, publ: 1 Mar 2018