An Efficient Next Word Prediction for Accurate Information using Deep Learning Algorithms

B.Tarakeswara Rao[1], E.Ramesh[2], A. Srinagesh[3] K.Srınıvasa Rao [4], N.Kıran Kumar[5], P. Siva Prasad[6], B.Naga Mallikarjuna[7], K.Arun[8]
[1]Professor, Dept. of CSE, KHIT, tarak7199@gmail.com
[2.] Assistant Professor, Dept. of CSE, R.V.R. & J.C College of Engineering, eramesh808@gmail.com
[3]Professor, Dept. of CSE, R.V.R. & J.C College of Engineering, asrinagesh@gmail.com
[4]Assistant Professor, Department of IT, Bapatla Engineering College,ksr2bec@gmail.com
[5]Assistant Professor, Department of CA, Bapatla Engineering College,kiran_kiron@hotmail.com
[6]Assistant Professor, Dept. of CSE, R.V.R. & J.C College of Engineering, prasadsiva_17@yahoo.com
[7]Assistant Professor, Dept. of Computer Science, Vignan Degree College, mallikharjunarao.b@gmail.com
[8]Assistant Professor, Dept. of CSE, Narasaraopeta Engineering College, karun014@gmail.com

Abstract:

Natural language processing and language models define subsequent phrase predictions. To bet, the following matching sentences are used in search engines, sentence or text content processing, and documentation applications. The most likely phrase is a high-value match for that sentence. In this task, subsequent phrase predictions are performed using the deep learning version. First, we preprocessed the text content, normalized the text content, and implemented four specific deep learning classifiers to experiment and check statistics for expecting subsequent words. Canonical Neural Network (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Bidirectional Long Short-Term Memory (BiLSTM). Of these deep algorithms, CNN when implemented contributed a high loss and much lower accuracy, and Bidirectional LSTMs resulted and were noted with high accuracy and low loss. These classifiers are run sequentially and comparisons are primarily based on loss discounts and accuracy characteristics. The results obtained show that the CNN's loss discount and accuracy were the worst and BiLSTM achieved the highest quality.

Keywords:
Natural Language Processing, Deep Learning, Prediction, CNN, LSTM, BiLSTM

1. Introduction

Text mining has gained quite a significant importance during the past few years. Data in recent times are available to users through many sources like electronic media, digital media, and many more. This data is usually available in the most unstructured form and there exist a lot of ways in which this data may be converted into a structured form. In many real-life scenarios, it is highly desirable to classify the information in an appropriate set of categories. News contents are one of the most important factors that influence various sections.

The objective of this paper is to efficiently classify web news into the specified four categories like health, business, entertainment, and science & technology. In order to achieve this initially the Natural Language Processing techniques are applied to get the interesting pattern and efficient Machine Learning classification algorithms are applied like SVM, LSTM, Decision Tree, and KNN thus high accuracy is expected to be obtained.

The contributions of the proposed framework are threefold:

1) The proposed framework can retrieve latent semantic information about social entities from the big data of media texts.

2) The proposed framework makes it possible to assess the soft power of social entities based on analyzing and projecting the media image of these social entities, which is constructed out of related media texts, to the soft power space.

3) The proposed framework integrates the top-down, deductive, human-intelligence based, and useful.

In this paper, we have considered the problem of the classification of news articles. This paper presents algorithms for category identification of news and has analyzed the shortcomings of a number of algorithmic approaches.

2. Review of Literature

Language modeling represents the first boundary in neural network research. A significant benchmark for Neurolinguistic modeling is Bengio et al. [1], who implemented the n-gram language model as a feed-forward neural network with historical words as input and predicted words as output. Schwenk et al. [2] take these language models into machine translation (also known as "continuous spatial language models") and use them in reclassification, similar to previous work on speech and voice recognition. Schwenk [3] suggested a number of changes and implementation is carried out using an open-

Manuscript received June 5, 2022
Manuscript revised June 20, 2022
https://doi.org/10.22937/IJCSNS.2022.22.6.83

source tool kit. Schwenk [4], developed and carried out an analysis based on GPU training (Schwenk et al [5], Finch et al. [6] used an iterative neuronal network language template to re-evaluate the best list for the translation system.

Sundermeyer et al. [7] compared the feed-forward implementation with long short-term neural network language models; Mikolov [8] proposed significant improvements by re-ranking n-best lists of machine translation systems with a recurrent neural network language model.

Mikolov [8] reported significant improvements with a reordering of the list of the best machine translation systems with the cyclic neural network language model by first grouping words into classes and encoding words as a class- and word-in-class bit pairs,

Baltescu et al. [10] sufficiently reduced computing complexity to enable the integration of the neural network language model into the decoder. Another way to reduce the complexity of the calculation to allow for the integration of the decoder is to use the contrast and noise estimate of Finch [9]. It standardizes the output score of the template during training, eliminating the need to calculate values for all possible output words. Wang et al. [11] converted a continuous spatial language model from a limited list of 8,192 words to a traditional n-gram language model in the ARPA (SRILM) format.

Wang et al. [11] presented a method to merge (or "zoom") a continuous-spatial language model with a traditional n-gram language model to take advantage of the two best guesses for words in a list drawn with a compact and complete enhancement of the traditional model. Neurolinguistic models are not deep learning because they use many hidden layers. Wang et al. [12] presented a novel method of “merging” or “growing” a continuous space language model with a traditional n-gram language model to take advantage of both better estimates for the words in the shortlist and the full coverage from the traditional model.

However, Luong et al. [13] showed that between three and four masked coatings improved after conventional coating. Terminal neural machine translation becomes more difficult.

A data-oriented monolingual language model is added by Gülcehre et al. [14]to this model, in the form of a cyclic neural network operating in parallel. They compared the use of the language model re-arrangement (or re-organization) to a deeper integration where a self-contained unit governs the relative contributions of the language model and the translation model when predicting a word.

Baltescu and Blunsom [15] compared two classes that are based on word coding techniques with normalized scores with noise contrast, and estimates without normalization scores and showed that the latter gives better performance with higher speed and much higher accuracy in another way to enable simple decoder integration. Some internal representation studies focus only on language models.

Linzen et al.[16] proposed the subject-verb agreement task, especially when interrupted by other nouns, as a challenge for sequential models that should preserve agreement information. Gulordava et al. [17] extended this idea to several other hierarchic language issues.

Giulianelli et al. [18] construct a classifier to predict verb agreement information from internal states in different classes of the LSTM language model and go further and demonstrate that changing the state of decoding aggregates based on the information obtained through the classifier which allows making better decisions. He compared the quality of purely (variable) attentional models with recurrent neural networks involved in hierarchically dependent decisions. Their experiments show that the cyclic neural network best performs tasks like the subject-verb agreement limited by a recursive phrase.

Zhang and Bowman[19] show that states obtained from a two-dimensional language model are better for parts of speech marking and hyper-score tasks than encoder states from a neural translation model.

Dhar and Bisazza [20] explored whether multilingual training leads to more general syntactic generalization, but could achieve find only a small improvement in agreement tasks when completely isolated vocabularies.

3. Methodology of the Proposed Work:

Why use LSTM?

Vanishing gradient descent is a problem faced by neural networks when we go for backpropagation. It has a huge effect and the weight update process is widely affected and the model becomes useless. This paper deals with how we can use a neural model better than a basic RNN and use it to predict the next word. We deal with a model called Long Short term Memory (LSTM).

Long Short-Term Memory method: In LSTM, it focuses in the classification of text where an identified classifier

can learn long-term dependencies between the texts. The LSTM classifier is a form of recurrent neural network or RNN, which is a layered network that uses the previous outputs for the inputs of the next layer.

$St, 1 = S1(St - 1, 1, Xt)$ First Layer --- (3.1)

$St, i = Si(St - 1, i. St, i - 1)$ (3.2)

$Yt = Si + 1(St, I)$ (3.3)

$Y^{\sim}\text{-}Y_{t=}\text{loss}$ (3.4)

Feedback connections are included in LSTM architecture, permitting it to implement with sequences of data as a replacement in place of just single data points. Therefore an LSTM node consists of a cell, input gate, output gate, and forget gate. The cell is what remembers the values over a time interval and the three gates regulate how the informal- tin will flow through the cell.

$S_t = S_{\sigma g}\, W_f\, Xt + U_f S_{t-1} + b_f$ (3.5)

$S_{ft} = S_{\sigma g}(W_i Xt + U_i S_{t-1} + bi)$ (3.6)

$S_{ot} = S_{\sigma g}(W_o Xt + U_o S_{t-1} + b_o)$ (3.7)

$S\tilde{}_t = S_{\sigma h}\,(W_c xt + U_c S_{t-1} + b_c)$ (3.8)

$S_t = St \circ c_{i-1} + it \circ c\tilde{}_t$ (3.9)

$S_t = S_{ot} / \sigma_h\,(c_t)$ (3.10)

Equations (3.5)–(3.10) which are mentioned above are used in the creation of an LSTM with a forget gate. Where W and U are matrices containing the weights for the inputs and recurrent connections. Xt is the input vector unit, S_{ft} is the for-get activation vector, it is the input activation vector, ot is the output activation vector, St is the output vector unit, c˜t is the cell input activation vector and ct is the cell state vector. σg and σh are the activation functions of sigmoid and hyperbolic tangents, respectively.

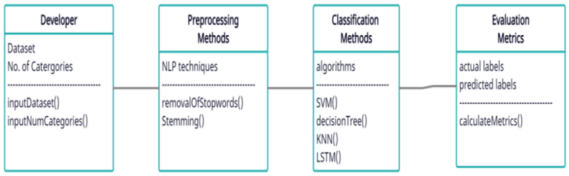


Fig: 3.1 Architecture of the Proposed Techniques

As shown above, the proposed framework is compatible with any word-embedding technology as long as its extracted word vectors of keywords can accurately represent the semantics of these keywords. There are various word-embedding technologies available, such as word2vec, Glove, ELMo, BERT, and GP. We can use the TensorFlow library in python for building and training the deep learning model.

MODULES IMPLEMENTED

1. Data pre-processing
2. Prediction
3. Classification and Evaluation

For example,

Matching datasets and tokenizers

Step 1: Preprocessing

Step 2: Post-processing

Word2Vec tokenization:

Case 0: Words in the dataset and the dictionary
Case 1: Words not in the dataset or the dictionary
Case 2: Noisy relationships
Case 3: Rare words
Case 4: Replacing rare words
Case 5: Entailment
Result: Predicted Word

Algorithm:

```
Class Recurrent Neural Network _RNN(library.nn.Module)
def_init_(self,vocab_size,hidden_size)
super(RNN,self).init_
self.hidden_size=hidden_size
self.embedding=library.nn.module(Vocab_size,hidden_size)
self.gru=library.nn.GRU(hidden_size,hidden_size)
self.out=library.nn. Linear(hidden_size,vocab_size)
self.softmax=library.nn.LogSoftmax(dim=1)
def self(input,hidden,vocab)
embedded=self.embedding(library.tensor([input])),view(1,1,-1))
output,hidden=self.gru(embedded,hidden)
output=self.softmax(self.out(self[0])
return output,hidden
def_init(hidden,self)
return.library.zeros(1,1,self.hidden_size)
```

Similarly, Bidirectional LSTM (BiLSTM) is a recurrent neural network used primarily in natural language processing. Unlike standard LSTM, the input flows in both directions, and it's capable of utilizing information from both sides.

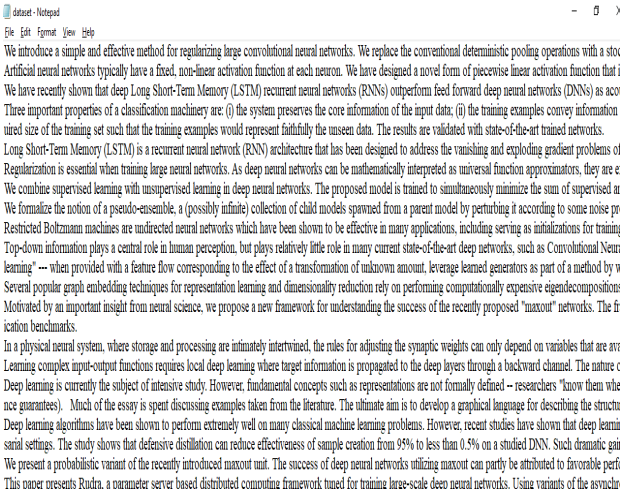
4. Experiments and Results

The dataset was trained and tested through the four deep learning algorithms, namely CNN canonical neural networks, RNN recurrent neural network, LSTM Long Short-Term Memory, and BiLSTM Bidirectional long-short Memory. Models are created with input, output, and hidden layers. Before using deep learning algorithms, initially pre-processing the dataset, partitioning the dataset, and applying the word embedding is performed on the

dataset, and a universal embedding model is used in this approach.

4.1 Dataset and Preprocessing:

The dataset collected from the Stanford-TensorFlow-tutorials1, the dataset is a collection of 7200 abstracts of the various research papers in computer science, machine learning, deep learning, computer networks, and research areas in computer science. The dataset contains 79,776 lines and 1,108,656 individual words. This dataset can be used to predict computer science-related words. Preprocessing of the dataset is used to prepare the dataset in the required form, in the abstract of the paper some punctuation symbols, numbers, abbreviations, and some special symbols were presented, all such types of unnecessary information were cleared with the pre-processing module.


dataset - Notepad

File Edit Format View Help

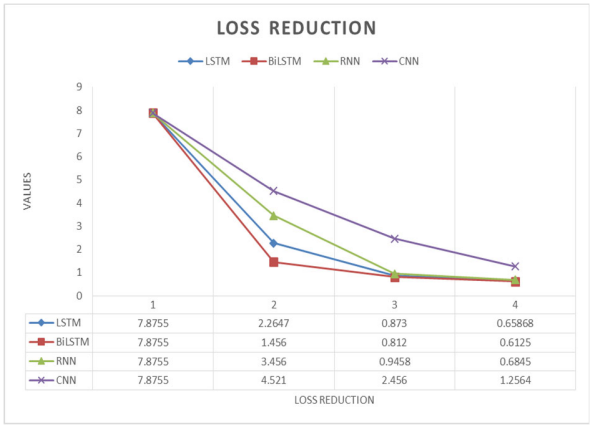
We introduce a simple and effective method for regularizing large convolutional neural networks. We replace the conventional deterministic pooling operations with a stoc
Artificial neural networks typically have a fixed, non-linear activation function at each neuron. We have designed a novel form of piecewise linear activation function that i
We have recently shown that deep Long Short-Term Memory (LSTM) recurrent neural networks (RNNs) outperform feed forward deep neural networks (DNNs) as acou
Three important properties of a classification machinery are: (i) the system preserves the core information of the input data; (ii) the training examples convey information
uired size of the training set such that the training examples would represent faithfully the unseen data. The results are validated with state-of-the-art trained networks.
Long Short-Term Memory (LSTM) is a recurrent neural network (RNN) architecture that has been designed to address the vanishing and exploding gradient problems of
Regularization is essential when training large neural networks. As deep neural networks can be mathematically interpreted as universal function approximators, they are e
We combine supervised learning with unsupervised learning in deep neural networks. The proposed model is trained to simultaneously minimize the sum of supervised an
We formalize the notion of a pseudo-ensemble, a (possibly infinite) collection of child models spawned from a parent model by perturbing it according to some noise pr
Restricted Boltzmann machines are undirected neural networks which have been shown to be effective in many applications, including serving as initializations for training
Top-down information plays a central role in human perception, but plays relatively little role in many current state-of-the-art deep networks, such as Convolutional Neura
learning" -- when provided with a feature flow corresponding to the effect of a transformation of unknown amount, leverage learned generators as part of a method by w
Several popular graph embedding techniques for representation learning and dimensionality reduction rely on performing computationally expensive eigendecompositions
Motivated by an important insight from neural science, we propose a new framework for understanding the success of the recently proposed "maxout" networks. The fi
ication benchmarks.
In a physical neural system, where storage and processing are intimately intertwined, the rules for adjusting the synaptic weights can only depend on variables that are ava
Learning complex input-output functions requires local deep learning where target information is propagated to the deep layers through a backward channel. The nature o
Deep learning is currently the subject of intensive study. However, fundamental concepts such as representations are not formally defined -- researchers "know them whe
nce guarantees). Much of the essay is spent discussing examples taken from the literature. The ultimate aim is to develop a graphical language for describing the structu
Deep learning algorithms have been shown to perform extremely well on many classical machine learning problems. However, recent studies have shown that deep learni
sarial settings. The study shows that defensive distillation can reduce effectiveness of sample creation from 95% to less than 0.5% on a studied DNN. Such dramatic gai
We present a probabilistic variant of the recently introduced maxout unit. The success of deep neural networks utilizing maxout can partly be attributed to favorable perfo
This paper presents Rudra, a parameter server based distributed computing framework tuned for training large-scale deep neural networks. Using variants of the asynchr

Fig 4.1: sample dataset

Layers were added to this model which is required to perform the deep learning approach. Using the mentioned algorithms the dataset was trained and tested, which contains four epoch stages. Figure 4.2 the first epoch from 1 to 50, second epoch 1-80, third epoch 1-120, and final epoch 1-200 four stages are verified. BiLSTM recorded the loss as very low, which is the best model. Figure 4.3 explains the sample training module with epochs up to the 200 range. LSTM and BiLSTM modules get the lowest loss values, BiLSTM reaches the very lowest value of the loss and it improved the accuracy of the method in this module.

```
61/61 [==============================] - ETA: 0s - loss: 7.8755 - accuracy: 0.0023
Epoch 1: loss improved from inf to 7.87548, saving model to best_model.hdf5
61/61 [==============================] - 23s 290ms/step - loss: 7.8755 - accuracy: 0.0023 - lr: 0.0010
Epoch 2/200
61/61 [==============================] - ETA: 0s - loss: 7.8635 - accuracy: 0.0044
Epoch 2: loss improved from 7.87548 to 7.86345, saving model to best_model.hdf5
61/61 [==============================] - 18s 291ms/step - loss: 7.8635 - accuracy: 0.0044 - lr: 0.0010
Epoch 3/200
61/61 [==============================] - ETA: 0s - loss: 7.8146 - accuracy: 0.0031
Epoch 3: loss improved from 7.86345 to 7.81459, saving model to best_model.hdf5
61/61 [==============================] - 18s 290ms/step - loss: 7.8146 - accuracy: 0.0031 - lr: 0.0010
Epoch 4/200
61/61 [==============================] - ETA: 0s - loss: 7.5868 - accuracy: 0.0028
Epoch 4: loss improved from 7.81459 to 7.58675, saving model to best_model.hdf5
61/61 [==============================] - 18s 290ms/step - loss: 7.5868 - accuracy: 0.0028 - lr: 0.0010
Epoch 5/200
61/61 [==============================] - ETA: 0s - loss: 7.3798 - accuracy: 0.0028
Epoch 5: loss improved from 7.58675 to 7.37976, saving model to best_model.hdf5
61/61 [==============================] - 17s 286ms/step - loss: 7.3798 - accuracy: 0.0028 - lr: 0.0010
Epoch 6/200
61/61 [==============================] - ETA: 0s - loss: 7.2107 - accuracy: 0.0026
Epoch 6: loss improved from 7.37976 to 7.21069, saving model to best_model.hdf5
61/61 [==============================] - 18s 288ms/step - loss: 7.2107 - accuracy: 0.0026 - lr: 0.0010
Epoch 7/200
61/61 [==============================] - ETA: 0s - loss: 7.1184 - accuracy: 0.0021
Epoch 7: loss improved from 7.21069 to 7.11842, saving model to best_model.hdf5
61/61 [==============================] - 17s 285ms/step - loss: 7.1184 - accuracy: 0.0021 - lr: 0.0010
```

Fig: 4.2 Sample output of Epoch=200 for loss reduction



	1	2	3	4
LSTM	7.8755	2.2647	0.873	0.65868
BiLSTM	7.8755	1.456	0.812	0.6125
RNN	7.8755	3.456	0.9458	0.6845
CNN	7.8755	4.521	2.456	1.2564

Fig: 4.3 Loss reduction comparisons with deep learning algorithms

5. CONCLUSION

To guess the new word in a sentence is important in word guessing, word suggestions, or search engine applications. In this paper, to guess words, a dataset was collected which contains relevant semantic words so as fit a given sentence. We experimented with different machine learning classifiers that demonstrated more accurate results with less time and space complexity for web applications based on textual data. As per the results obtained, we have compared four classifiers K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree (DT), and

Long Short-Term Memory (LSTM). SVM is coming out on top with the best accuracy of 95.04% and KNN with the worst at 88.72%. The results of the top classifiers demonstrated good accuracy, a reduction in time needed for the training and testing phases of classification (time complexity and a reduction of total space complexity).

In order to decrease the time and space complexity, several NLP techniques are used and the accuracy that was achieved was good and acceptable, Thus the latter brings it easy for giving effective predictions to the user, and our approach offers directions for further improvements using BERT or Transformers based models.

6. References

[1] Bengio, Yoshua, et al. "Out-of-sample extensions for lle, iso map, mds, eigenmaps, and spectral clustering." Advances in neural information processing systems 16 (2003).

[2] Schwenk, Holger, Daniel Déchelotte, and Jean-Luc Gauvain. "Continuous space language models for statistical machine translation." Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions. (2006).

[3] Schwenk, Holger. "Continuous space language models." Computer Speech & Language 21.3 492-518. (2007)

[4] Schwenk, Holger. "Continuous space language models." Computer Speech & Language 21.3 (2007): 492-518.

[5] Schwenk, Holger. "Continuous-Space Language Models for Statistical Machine Translation." Prague Bull. Math. Linguistics 93 (2010): 137-146.

[6] Schwenk, Holger, Anthony Rousseau, and Mohammed Attik. "Large, pruned or continuous space language models on a GPU for statistical machine translation." Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT. 2012.

[7] Sundermeyer, Martin, et al. "Comparison of feedforward and recurrent neural network language models." 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, 2013.

[8] Mikolov, Tomas, and Geoffrey Zweig. "Context-dependent recurrent neural network language model." 2012 IEEE Spoken Language Technology Workshop (SLT). IEEE, (2012).

[9] Finch, Andrew, Paul Dixon, and Eiichiro Sumita. "Re-scoring a phrase-based machine transliteration system with recurrent neural network language models." Proceedings of the 4th Named Entity Workshop (NEWS)(2012).

[10] Chen, Xie, et al. "Efficient GPU-based training of recurrent neural network language models using spliced sentence bunch." Fifteenth Annual Conference of the International Speech Communication Association. 2014.

[11] Seltzer, Michael L., Dong Yu, and Yongqiang Wang. "An investigation of deep neural networks for noise robust speech recognition." 2013 IEEE international conference on acoustics, speech, and signal processing. IEEE, 2013.

[12] Liu, Xunying, et al. "Efficient lattice rescoring using recurrent neural network language models." 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2014.

[13] Luong, Minh-Thang, Richard Socher, and Christopher D. Manning. "Better word representations with recursive neural networks for morphology." Proceedings of the seventeenth conference on computational natural language learning. 2013.

[14] Gulcehre, Caglar, et al. "On using monolingual corpora in neural machine translation." arXiv preprint arXiv:1503.03535 (2015).

[15] Baltescu, Paul, and Phil Blunsom. "Pragmatic neural language modeling in machine translation." arXiv preprint arXiv:1412.7119 (2014).

[16] Linzen, Tal, Emmanuel Dupoux, and Yoav Goldberg. "Assessing the ability of LSTMs to learn syntax-sensitive dependencies." Transactions of the Association for Computational Linguistics 4 (2016): 521-535. (2016)

[17] Aina, Laura, Kristina Gulordava, and Gemma Boleda. "Putting words in context: LSTM language models and lexical ambiguity." arXiv preprint arXiv:1906.05149 (2019).

[18] Giulianelli, Mario, et al. "Under the hood: Using diagnostic classifiers to investigate and improve how language models track agreement information." arXiv preprint arXiv:1808.08079 (2018).

[19] Zhang, Kelly W., and Samuel R. Bowman. "Language modeling teaches you more syntax than translation does Lessons learned through auxiliary task analysis." arXiv preprint arXiv:1809.10040 (2018).

[20] Dhar, Prajit, and Arianna Bisazza. "Does syntactic knowledge in multilingual language models transfer across languages?." Proceedings of the 2018 EMNLP workshop BlackboxNLP: analyzing and interpreting neural networks for NLP. 2018