Machine Learning Models For Predicting And Modeling More Engaging Business Manuscripts

Aakarsh Sinha
asinha16@earlham.edu
Earlham College
Richmond, Indiana

ABSTRACT

Given the high stakes and intense competition within all areas of the industry, intelligent business decisions are more important than ever. Data analysis plays an important role as a critical strategic weapon in business. Neural networks are one of the most powerful machine learning algorithms that can model and solve approximation, optimization, prediction, and classification problems. However, even though tools that use neural networks and machine learning to analyze and predict outcomes have been extensively researched, there is a lack of research on business applications built around sequence to sequence models.

This paper presents two machine learning Long Short-Term Memory (LSTM) encoder-decoder models that take in text-based advertisements as input, and analyze and restructure them to make them more engaging and readable to the target audience. The model is trained on a dataset created by scraping advertisements from Groupon.com which are analyzed and scored using Coleman-Liau Readability Formula, The Flesch Reading Ease formula, and a dataset of active call-to-action verbs.

KEYWORDS

Neural Networks, Convolutional Layers, Text Analysis, Machine Learning, Direct Marketing, Prediction, Sentiment Analysis

1 INTRODUCTION

As the field of computer science has advanced, new techniques and algorithms such as neural networks have been developed that allow a computer to be trained to recognize hidden patterns and correlations in raw data. Neural networks try to emulate or model how the human brain processes information by emulating the biological structure of the human brain. In order to do this, neural networks mimic certain aspects of the physical structure of the brain with a web of neural connections. Neural networks are massively parallel, fast, and fault-tolerant, and can model phenomena that are difficult to describe. Artificial neural networks are also capable of performing tasks and modeling phenomena that are difficult to describe, such as approximation, optimization, classification, prediction, generalization, relation, and abstraction, which can be extremely beneficial for businesses [7].

Neural networks are non-linear and extremely useful to model practical situations since they make no assumptions about the distribution of properties of the data [11]. Neural networks can perform complex tasks by learning through experience and examples by extracting essential characteristics from given data. Neural networks do not need to know the task they are going to perform in advance. They have the ability to provide highly accurate and robust solutions for complex non-linear tasks, such as fraud detection, business lapse/churn analysis, and risk analysis [15].

Neural networks have been used for a variety of applications such as objection recognition and speech recognition. However, sequence to sequence encoder-decoder models are relatively new, and were first proposed by Google in 2014 [12]. Most of the research around sequence to sequence models is focused on tasks in natural language processing such as language modeling and word embedding extraction [1]. However, little research exists around commercial applications of these sequence to sequence models as well as leveraging these models to benefit businesses.

Our project aims to use neural networks to improve the readability and engageability of text-based advertisements. This will help businesses create more engaging marketing material, which will, in turn, help businesses attract and retain more customers. In order to achieve these aims, the project will implement and analyze variations of multiple encoder-decoder models to learn how to form more readable and engaging text-based advertisements and to make them more engaging and readable while trying to maintain the semantic integrity of the advertisement.

This paper first discusses research in text analysis, sentiment classification, and sequence to sequence translation that have helped mold the initial foundation of our project. The design, framework, and detailed implementation of the project are then further discussed, which includes the creation and analysis of the dataset used, as well as the testing methodologies that have and will be used to assess the effectiveness and accuracy of the models used in our project. The next steps that will be taken as a part of this project are then discussed as well as future work that can potentially be done to improve the accuracy of the model.

The following are the major contributions of this research project:

- The web scraping tool used to scrape advertisements using the Groupon API.
- An analysis tool that uses Coleman-Liau Readability Formula, The Flesch Reading Ease formula and a dataset of

Earlham College Computer Science Senior Capstone

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

active call-to-action verbs to analyze, score, and create a weighted dataset of advertisements.

- A web-based application that uses the proposed neural network model to restructure business sentences to be more engaging.
- Machine learning models to restructure non-engaging sentences into more engaging sentences.
- Analysis of variations of multiple encoder-decoder models.

2 RELATED WORK

This section includes papers that use neural networks to extract features from sentences, analyze them, and capture semantic relationships between them for various applications. Capturing semantic relationships is pertinent to our project in order to conserve the semantic structure of the input sentences while restructuring it to make it more engaging. This section is divided into three subsections - text analysis, sequence to sequence translations, and sentiment classification.

2.1 Text Analysis

Neural networks have been widely used to extract relevant features from text in order to analyze them and capture connections and relationships between them in order to understand a language. This subsection talks about research that analyzes text using convoluted neural networks.

Kalchbrenner et al. [5] discussed a Dynamic Convolutional Neural Network that is used to semantically model sentences. The neural network handles input sentences of varying lengths and is capable of explicitly capturing short and long range relations within the sentence structure. The authors used filters at higher layers in order to capture semantic relations between non-continuous phrases that are far apart in the input sentence.

Santos and Gatti [4] presented a neural network, named Character to Sentence Convolutional Neural Network (CharSCNN). CharSCNN is used for analyzing short texts. It used two convolutional layers in order to extract relevant features from the input. The first layer of the network converted words to vectors. The vector was made up of two sub-vectors called embeddings. The first "Word-level embeddings" captured syntactic and semantic information while the second "character-level embeddings" captured morphological and shape information. This second layer then produces local features around each word in the sentence to form relationships amongst them.

Ke and Hagiwara [6] proposed a neural network that can automatically learn, retain the information, and answer questions from a given piece of text. The sentences were then broken down into clauses. The neurons of the clauses were linked to the sentences containing them. Similarly, phrases, clauses, and words were then linked to the corresponding phrases.

While papers [4–6], all used convolutional layers to perform analysis, papers [5] and [6] were distinctly similar in structure. Paper [5] included a layer for characters while paper [6] used a concept layer. The usage of the concept layer was very interesting, as it allowed [6]'s neural network to respond correctly even when no direct answer or relationship was found in the corpus. The usage of a concept layer in our project similar to that of [6] might also help in increasing the accuracy of the output. Additionally, the concept layer model can be further combined with Kalchbrenner et al.'s technique to improve upon the accuracy and breaking of the different word categories in order to be further categorized. The primary purpose of [6] was to answer questions from a given piece of text and use very detailed layers in order to extract information and form relationships between sentences, something that our project will have to heavily rely on in order to accurately predict a better way to write sentences.

2.2 Sequence To Sequence Translation

Cho at al. [2] proposed a novel neural network model that used an encoder and a decoder to encode a sequence of symbols into a fixed length vector representation, and then decode the representation into another sequence of symbols. The encoder and decoder of the proposed model are jointly trained to maximize the conditional probability of a target sequence given a source sequence. The authors were able to show that the new model is able to accurately capture linguistic regularities in the phrase pairs and is also able to propose well-formed target phrases.

Zhang et al. [16] introduced a deep LSTM-based natural language processing model that encodes sentences while retaining semantic information with similar vector representations. The vector representation is first extracted from an encoder-decoder model which is then trained on paraphrased pairs to learn dependencies over long sentences. The proposed model is then used for sentence paraphrasing and paragraph summarization.

Patro et al. [9] introduced a sequential encoder-decoder model for obtaining sentence-level embedding and generating paraphrases that semantically resemble the original sentence. The authors use a one-hot vector representation for every word and obtain a word embedding that is then fed to an LSTM-based encoder which encodes the features of the sentence. The model also used a sequential pairwise discriminator that evaluates the sentences for authenticity and is trained with a loss function that penalizes paraphrased sentence embedding distances if they are too large.

Wang et al. [14] used an encoder-decoder based LSTM model to simplify text. The authors use three different vocabularies in their experiments and reversed, sorted, and replaced words in the input sequence to simulate changing the sentence structure, replacing words, and removing words. The results showed that the LSTM Encoder-Decoder is able to learn the reversing, sorting, and replacement operation rules from the provided data, and thus has the potential to simplify a complex text.

[2, 9, 14, 16] all use an encoder-decoder model to form semantic relationships within sentence structures. While they all use somewhat similar encoder-decoder models, the model presented in [9] achieved a higher accuracy than others because of the usage of a discriminator in addition to their basic encoder-decoder model. Our project uses a variation of these encoder-decoder models to learn the contextual and semantic meaning of words in the sentences in order to preserve meaning while converting non-engaging documents to engaging documents. Our project also uses a one hot vector encoding - similar to that used by [9] to obtain word embedding for the encoder-decoder model.

2.3 Sentiment Classification

This subsection discusses neural networks that are used to perform sentiment analysis on a given dataset.

Aliaksei and Alessandro [10] presented a neural network that used a sentence matrix for each input in which words were represented by distributional vectors. For each input, a sentence matrix was built. The authors used a unique three-step process to train the neural network. In the first step, a "word2vec" model was used to learn word embeddings in an unsupervised corpus. Word2vec is an acclaimed model that can make highly accurate guesses about a word's meaning based on past appearances [8]. In order to accomplish this, word2vec uses vectors as numerical representations of word features, which are then used to establish a word's association with other words in order to perform sentiment classification.

Tang et al. [13] used four large-scale review datasets from IMDB and Yelp in order to perform document-level sentiment classification. The research used a typical convolutional layer model to learn sentence representation. It also used a unique gated recurrent neural network model.

[10] used a convolution neural network whereas [13] used a gated recurrent network to classify sentiments within sentences. While gated recurrent neural network have extensively been used in sentiment classification, as they can be trained to retain data for long periods of time helping them perform well in tasks that require capturing long-term dependencies [3], the usage of a "word2vec" to adaptively encode the semantics of sentences and their relations model helped [10] achieve a higher accuracy than that of [13].

3 DESIGN AND IMPLEMENTATION

The framework of our project uses a model view controller (MVC) structure and is shown in Figure 1. Following the MVC architectural pattern, the project is divided into two interconnected elements - the frontend and the backend. The project uses Keras, an open-source neural-network library written in Python with a Tensorflow backend to create the neural network. ReactJS, a JavaScript library maintained by Facebook, is used to create the client-facing frontend of the web application, which allows the user to upload text-based advertisements as input and get a more engaging advertisement as output. Flask, a Python framework, is used for creating the middleware that handles Hypertext Transfer Protocol (HTTP) requests to and from the frontend.

The backend handles the creation, analysis and pre-processing of the dataset. It also contains the neural network as well as the middleware that contains GET and POST APIs that handle HTTP requests that take in advertisements from the frontend as input and send back a more readable and engaging advertisement back to the frontend as output.

Once the dataset is scraped, analyzed, and created, it is sent to the neural network for training. Once trained, the neural network then accepts advertisements from the frontend (through the middleware) and produces a more readable and engaging version of it. Our project intends to first create a basic encoder-decoder model of a neural network and then build upon it to implement and analyze variations of it. The implementation of the first version of the encoder-decoder model is discussed in the subsections below.

3.1 Dataset

This section discusses how the dataset used to train the neural network is acquired, analyzed, and pre-processed.

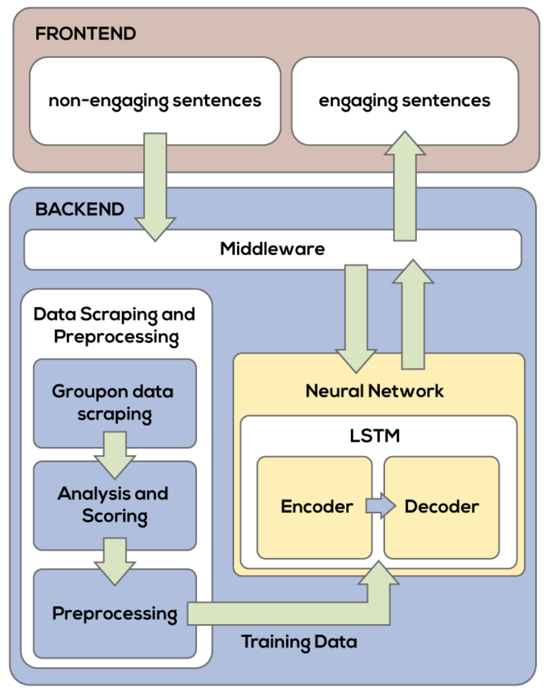

Figure 1: Framework Of The Project.

3.1.1 Creating And Weighing The Dataset

Advertisements from Groupon.com are first downloaded using APIs provided by Groupon.com that allow developers to access advertisements posted on their platform. A Python script is then used to create a CSV file from the advertisements that are scraped using the API. After the CSV file has been created, it is then analyzed using two readability formulas.

- Coleman-Liau Readability Formula
- The Flesch Reading Ease formula

After it has been analyzed and scored using the formulas, the data is then separated into buckets of readability. The bucket with the highest number of items is considered readable by most audiences and is used to create a new dataset. This new dataset is then compared to a dataset of active call-to-action words such as "Book now" and "Buy Now". Scores are assigned according to Table 1. The highly engaging items in the dataset are then separated from the dataset to create a new dataset of readable and engaging marketing materials.

3.1.2 Preprocessing The Data

"<Start>" and "<End>" markers are then added to the beginning and end of each advertisement in the newly created dataset of engaging

Type	Description	Score
Highly Engaging	One or more call-to-action words	1
Non-Engaging	No call-to-action words	0

Table 1: Score Assignment Model.

and readable marketing materials to help our model understand where the engaging sentences start and where they end.

3.2 Frontend

The frontend serves as the entry point of the input that is to be improved. Once the user uploads the advertisement, a POST API call is made which sends the advertisement to the backend where it is processed.

3.3 Backend

The backend contains the neural network that has been trained to produce more engaging/readable sentences based on the dataset of engaging sentences.

The backend consists of two main components - the API gateway and the neural network. The API gateway handles the API calls and contains GET and POST methods that facilitate the transfer of data to and from the backend. The neural network processes the input advertisement and forms a new, more engaging advertisement which is then sent back to the frontend using another POST API call to be viewed by the user.

3.4 The Two Models

Two models have been used to translate and modify non-engaging and readable sentences to engaging and readable sentences. The first model is a basic encoder-decoder that relies solely on the learned semantics within engaging sentences. It encodes the advertisement into vectors and then pieces it back together, learning the structure of the engaging sentences and in turn restructuring and modifying non-engaging sentences through what it has learned. The second model uses two encoder-decoder models trained on two different datasets and relies on both engaging as well as non-engaging sentences to train map the features of engaging sentences on to non-engaging sentences.

3.5 First Model

The model uses Keras API with Tensorflow to create the neural network. In order to transform sentences to make them more engaging and readable a sequence to sequence (seq2seq) based learning model is being used. Seq2seq learning models are often used to convert sequences from one domain to another, for example, changing sentiment of a given sentiment to another or translating between languages. This seq2seq learning model is made up of a recurrent neural network (RNN) model that uses a long short-term memory (LSTM) architecture. RNNs use sequential information to remember previously captured informational data. This allows RNNs to recurrently predict new data based on relationships formed in the past. An example of this would be predicting what word comes next in a sentence by learning what usually goes before that word.

However, RNNs have short term memory which limits their ability to carry information from more than a few time steps. LSTMs help the model retain information for longer periods of time which helps in obtaining more accurate predictions.

The RNN is made up of an encoder-decoder model as shown in Figure 2. Encoder-Decoder models have been widely used in seq2seq applications that involve predicting a text output given a text input. The encoder and the decoder are both neural networks, the encoder takes in a sequence of text as an input and outputs vectors, the decoder takes in the encoder's output and tries to reconstruct it.

During the training phase, the neural network is fed sentences of advertisements from the engaging and readable dataset. The model learns the semantics between words in these sentences through a method called "teacher forcing" which forces the decoder to generate sequences, and compare it to what the correct output should be which is then used to restructure non-engaging sentences during the execution phase.

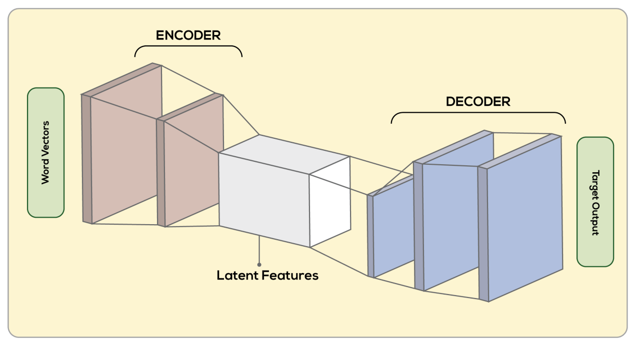

Figure 2: Architecture Of The First Model.

3.5.1 Training

During the training phase, the model tries to learn semantics between words and sentence structures using the encoder-decoder model, as discussed above.

3.5.1.1 Encoder Training

The encoder consists of two layers, the input layer and the LSTM layer. The input layers take in a matrix of one-hot-vectors and the LSTM layers contain hidden states which, during training, help the encoder learn how to reduce the input dimensions and encode representation.

Vector	***Timestep***	***Word***
[1,0,0,0]	Timestep 0	Buy
[0,1,0,0]	Timestep 1	The
[0,0,1,0]	Timestep 2	Pizza
[0,0,0,1]	Timestep 3	Now

Table 2: One-Hot-Vectors.

The input sentences are first assigned a one-hot vector encoding as shown in Table 2. Each of these vectors is then assigned a timestep. One-hot-vectors help represent words in a set of words. One (1) indicates the current word while zero (0) indicates all the other words.

3.5.1.2 Decoder Training

The decoder consists of three layers: the input layer, the LSTM layer, as well as the dense layer. The input layer takes in the output states of the encoder and passes it to the LSTM layer. The dense layers contain a softmax function that takes in the output of the LSTM layer as input and produces a probability distribution of words in its vocabulary. During training the decoder learns how to reconstruct the encoded representation to be as close to the original input as possible.

3.5.2 Optimizer And Loss Function

The model uses RMSProp as the optimizer which is an optimization algorithm that helps minimize the error rate of the model which is calculated by the loss function.

3.5.3 Execution

During this phase the model takes in the input sentence, but instead of using teacher enforcing, it uses the previously trained model to convert the input to a more engaging sentence. It does that by trying to restructure the input sentence using the semantic relationships between words in the more engaging and readable dataset it learnt during training.

3.5.4 Results

	Input
1	Equip your Mac with Windows software.
2	Click to claim your offers.
3	Use this guide featuring carefully selected deals from Amazon, starting at just $9.
	Output
1	Equip your Mac with Windows software today!
2	Click to claim your offers before they disappear!
3	Check out this guide featuring carefully handpicked deals from Amazon, starting at just $9.

Table 3: Inputs And Outputs Of The First Model.

The accuracy of the model represents how well the model can reproduce an engaging and readable sentence and is calculated by comparing the the similarities between the inputs and the outputs of the model during training. The accuracy of the model hovers around 90% during training (see Figure 3). It can accurately modify sentence structures it has seen during training (see Table 3) but is not able to accurately convert and modify advertisements that are not similar to the ones used to train the neural network.

During execution we use the average score of the readability formulas, the similarity between the input and the output as well as the number of call-to-action words contained within the modified advertisements to calculate the score. Table 3 presents a handful of conversions that the model has been able to efficiently translate.

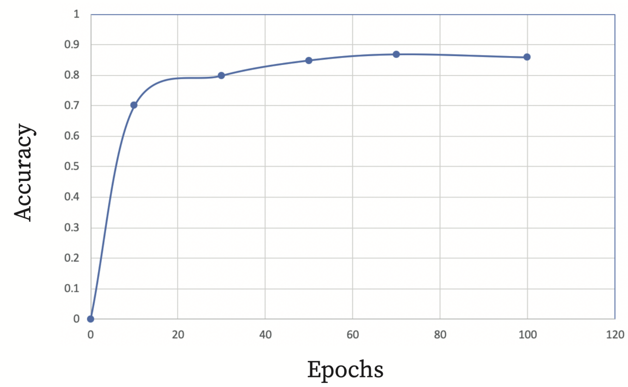

Figure 3: Training Accuracy.

3.6 Second Model

The second model uses two encoder-decoder models which have been illustrated in Figure 4. This second model incorporates an additional encoder-decoder model on top of the first model to train a secondary dataset of non-engaging/readable advertisements. The process of deciding the difference between non-engaging and highly-engaging marketing materials or sentences is discussed in section 4.1. The two encoder-decoder models are then trained, one on non-engaging/slightly engaging sentences and the other on the highly engaging sentences. The encoder of the first model and the decoder of the second model are then combined. The encoder of the first model is trained to encode the state of the input while preserving the contextual relationship of the words contained within highly engaging sentences. The output of this encoder is then used as the input of the trained decoder of the second model which is trained to restructure the non-engaging sentences without changing its meaning. This lets the model map the features of engaging sentences onto the non-engaging sentences.

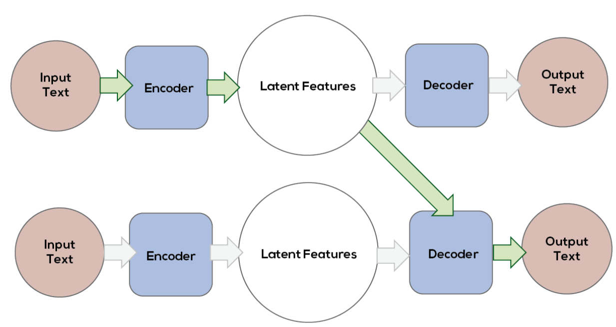

Figure 4: Architecture Of The Second Model.

3.6.1 Results

As the second model depends on similar sentence structures to map the features of engaging sentences on to the non-engaging sentences, it requires parallel data in order to efficiently function and output meaningful sentences. For example, the non-engaging sentence - *"Two-month membership to ABCmouse.com, providing kids ages 2-8 with a variety of educational activities, games & guided curricula."* and the highly-engaging sentence - *"Check out a huge selection of pet food and other essentials at Amazon."* have nothing in common which is why the outputs will not be legible. The inputs as well as the outputs of the second model are presented in Table 4.

	Input
1	Equip your Mac with Windows software.
2	Click to claim your offers.
3	Use this guide featuring carefully selected deals from Amazon, starting at just $9.
	Output
1	Equip Equip Equip Equip Equip Equip
2	Click claim claim claim.
3	Use Use Use Use this this this this

Table 4: Inputs And Outputs Of The Second Model.

4 CONCLUSION AND NEXT STEPS

We show two encoder-decoder LSTM models aimed at transforming text-based advertisements to make them more readable and engaging to the target audience. We first scrape advertisements from Groupon.com to create a dataset. We then analyze and score the dataset and then train our model using it.

Coleman-Liau Readability Formula and the Flesch Reading Ease formula are not accurate measures of engageability. They can calculate readability on a certain level but a dataset produced by a company - such as a dataset with real-world advertisements with click-through rates or rated by customers on its engageability would produce more accurate results. The accuracy of the model is calculated using what the output should look like. However, the final result is subjective to the end-user. In order to test the capability of the model, not only do we need to mathematically evaluate the efficiency of our model, we need to perform user testing. Additionally, using other variants of LSTM models such as bi-directional LSTM, reduced LSTM, or using an attention mechanism similar to that used by [16] present potential avenues of further research.

5 ACKNOWLEDGEMENTS

This project was supported by the Department of Computer Science at Earlham College and would not have been possible without the help, constructive feedback, and suggestions of Dr. Charles Peck, Dr. David Barbella, Dr. Xunfei Jiang, and Dr. Igor Minevich.

REFERENCES

[1] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Y. Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. (06 2014). https://doi.org/10.3115/v1/D14-1179

[2] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Y. Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. (06 2014). https://doi.org/10.3115/v1/D14-1179

[3] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Y. Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. (12 2014).

[4] Cícero dos Santos and Maíra Gatti. 2014. Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. Dublin City University and Association for Computational Linguistics, Dublin, Ireland, 69–78.

[5] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A Convolutional Neural Network for Modelling Sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Baltimore, Maryland, 655–665. https://doi.org/10.3115/v1/P14-1062

[6] Yuanzhi Ke and Masafumi Hagiwara. 2017. An english neural network that learns texts, finds hidden knowledge, and answers questions. *Journal of Artificial Intelligence and Soft Computing Research* 7, 4 (1 10 2017), 229–242. https://doi.org/10.1515/jaiscr-2017-0016

[7] Eldon Y. Li. 1994. Artificial neural networks and their business applications. *Information Management* 27 (1994), 303–313.

[8] Tomas Mikolov, Ilya Sutskever, Kai Chen, G.s Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. *Advances in Neural Information Processing Systems* 26 (10 2013).

[9] Badri N. Patro, Vinod K. Kurmi, Sandeep Kumar, and Vinay P. Namboodiri. 2018. Learning Semantic Sentence Embeddings using Sequential Pair-wise Discriminator. arXiv:cs.CL/1806.00807

[10] Aliaksei Severyn and Alessandro Moschitti. 2015. Twitter Sentiment Analysis with Deep Convolutional Neural Networks. 959–962. https://doi.org/10.1145/2766462.2767830

[11] Kate Smith-Miles and Jatinder Gupta. 2000. Neural Networks in Business: Techniques and Applications for the Operations Researcher. *Computers and Operations Research* 27 (09 2000). https://doi.org/10.1016/S0305-0548(99)00141-0

[12] Ilya Sutskever, Oriol Vinyals, and Quoc Le. 2014. Sequence to Sequence Learning with Neural Networks. *Advances in Neural Information Processing Systems* 4 (09 2014).

[13] Duyu Tang, Bing Qin, and Ting Liu. 2015. Document Modeling with Gated Recurrent Neural Network for Sentiment Classification. 1422–1432. https://doi.org/10.18653/v1/D15-1167

[14] Tong Wang, Ping Chen, Kevin Amaral, and Jipeng Qiang. 2016. An Experimental Study of LSTM Encoder-Decoder Model for Text Simplification. arXiv:cs.CL/1609.03663

[15] Bernard Widrow, David E. Rumelhart, and Michael A. Lehr. 1994. Neural Networks: Applications in Industry, Business and Science. *Commun. ACM* 37, 3 (March 1994), 93–105. https://doi.org/10.1145/175247.175257

[16] Chi Zhang, Shagan Sah, Thang Nguyen, Dheeraj Peri, Alexander Loui, Carl Salvaggio, and Raymond Ptucha. 2018. Semantic Sentence Embeddings for Paraphrasing and Text Summarization.