Generative adversarial networks for detecting Incongruence between News Headline and Body Text

Khoa Nguyen Le
University of Stavanger, Norway
k.lenguyen@stud.uis.no

ABSTRACT

The mis- or disinformation problems in the news titles are recently receiving much attention from organizations, governments, and scientific researchers. The incongruence between the headline and body of the news is a specific problem within this domain. Headlines make the initial and strong impression to the readers even after they read the whole article. Therefore, the incongruence detection is essential in helping the users to early identify a good article to read.

The incongruence problem should be tackled by a different approach from the methods applied to other problematic headlines such as clickbait or sensationalism. In this project, we experiment with Generative Adversarial Networks (GANs) to generate the headlines from body content then detect the incongruence by comparing the similarity between the original and the generated headlines. The experimental results show that this approach has the potential for improvements and new research directions in the future work.

KEYWORDS

generative, adversarial, networks, incongruence, news, headline, body text, bert, doc2vec

1 INTRODUCTION

With the current amount of information overload on the Internet, headlines play a very important role in helping people choose an appropriate article to consume before reading the entire content [10], and readers also remember the headline longer than the details in body text [8]. Thus, an incongruent headline would mislead people to exaggerated or false information.

The incongruence problem can be described by the following example from [8]:

Headline: *Air pollution now leading cause of lung cancer*

Body: *"We now know that outdoor air pollution is not only a major risk to health in general but also a leading* ***environmental*** *cause of cancer deaths." Dr. Kurt Straif, of IARC*

In the example above, missing "environmental" in the headline could mislead the readers by exaggerating the claim made in the body text. Air pollution is just the leading cause of environmental perspective and it may not be the leading cause that above all others. This headline, therefore, is not a proper representation of the news content. Detecting those misleading headlines could reduce the spread of false information.

One of the major obstacles when researching on this issue is the lack of large-scale public dataset [4]. Yoon et al. [33], in an attempt to build deep learning models to tackle the incongruence problem, have presented a method to create synthetic data based on real news articles. We then apply this method on the NELA-2017 dataset [14] to build data used for our model.

In some previous researches, the similarities between news headlines and body text have been used to detect the incongruent titles [9][30]. In this project, instead of using the article body directly to calculate the similarity score, we use *Generative Adversarial Networks* (GAN) to generate synthetic headlines from the body. The generated headlines will represent the news content and the similarity scores between the original headlines and the generated ones will be calculated. GAN was introduced in 2014 by Goodfellow et al. [12] and has received a lot of attention as well as inspired a huge number of other researches on natural language processing (NLP) or computer vision.

Before calculating the textual similarities, we build the embedding representations for both original and generated headlines using Doc2Vec [17] and Bidirectional Encoder Representations from Transformers (BERT) [7]. In our approach, the cosine similarity values are obtained from those embeddings. The embeddings are also used as the features for a simple binary classifier (linear classifier or SVM classifier) to predict the headline classes (congruent or incongruent). We also experiment with neural networks [26] as the classifier for comparison.

Experimental results show that the project brings some positive outcomes:

- We are, to the best of our knowledge, one of the early experiments that adopt GAN models to detect incongruent news titles.
- Recognizing the limitations of the existing model, paving the way for new approaches in the future.
- The method could be extended for related problems such as clickbait detection.
- Similarity scores between the original headlines and the generated headlines can be a useful feature for other classification supervised learning

2 RELATED WORK

There are many studies on *clickbait headlines* detection, a topic that is very closed to detecting incongruent headlines. Clickbait is a type of article titles that leave out the crucial information and force the readers to click to a link to find out the answer [3]. Clickbait detection faces the same problem of the lack of large-scale labeled training data. Kai Shu et al. [28] propose a framework, named as *Stylized Headline Generation (SHG)*, to generate synthetic headlines with specific styles from original documents with style transfer. They also use GANs and Variational Auto-Encoder in their model. The experiment results show that their solution is very promising

Supervised by Vinay Jayarama Setty.

Project in Computer Science (DAT620), IDE, UiS
2018.

for enlarging the clickbait dataset and improving the classification. However, it needs to be considered carefully if we want to apply this model for incongruence detection since the incongruent headlines lack the typical features of clickbait, they do not encourage the audience to click to the details of the story [4].

As one of the latest studies to tackle the incongruence problem, Yoon et al. [33] present two hierarchical neural network architectures that learn the textual relationship between the news headline and the body text. *Attentive Hierarchical Dual Encoder (AHDE)* encodes the body text from the word-level to the paragraph-level using two hierarchical RNNs. *Hierarchical Recurrent Encoder (HRE)* is simpler, it embeds each paragraph in the article content by averaging the word-embedding of the words in that paragraph. Then, applying a single RNN in paragraph-level to get the encoding vector of the whole body text. They also propose a method to produce the synthetic incongruence data which we use to build the dataset for this project's experiment from the real news articles. Both models they develop show decent performances compared to previous researches. We also apply their solution with our dataset as the baseline for our result comparison.

3 BACKGROUND

In this section, we introduce the theory in brief that is needed to build our approach.

3.1 Generative Model

Generative models G try to model the data distribution to generate examples of that data type. There are some different deep learning techniques used for text generation.

Recurrent Neural Networks (RNN) are a type of artificial neural networks (ANN) specifically used to find patterns in a sequence of data. RNN is basically a Long-Term memory based network. One disadvantage is that as the sequence grows larger, RNN gets difficulties to predict the next element in the sequence. Therefore, the generated text lack real meaning [15].

Long Short Term Memory (LSTM) is a special type of RNN. Unlike original RNN, it can capture the potential long-distance dependencies [6]. LSTMs have four neural network layers and they are, basically, uni-directional and receive information from past states only. [22]

Bi-directional LSTM learns from both past and future states. It consists of two hidden layers of uni-directional LSTM in opposite directions. This way, Bi-LSTMS are trained to predict both the next and previous values of the current state [31].

3.2 Discriminative Model

The discriminative model's target is to learn how to perform classification and regression on data.

Convolutional Neural Network (CNN) CNN has become popular when building a discriminative model. Yoon Kim [16] proposed the usage of CNN in the context of sentence classification, which was previously developed for computer vision. Even though CNNs may lose some vital context information, they still are a powerful tool for feature representation.

3.3 Reinforcement Learning

Reinforcement Learning (RL) is based on the concept of having an agent acting in a state space [1]. The target of the agent is to take actions that give the maximum reward based on its state. The agent uses a stochastic policy to make a decision on which actions to take. RL has been used commonly in the text generation models using GAN [35] [13].

3.4 Generative Adversarial Networks (GAN)

Generative Adversarial Networks (GAN) was introduced by Goodfellow et al in 2014 [12]. A GAN model consists of one *generator network G* that is trained to generate realistic samples and one *discriminator network D* to detect if a given sample is from the training data or generated by G. The model can be simply described by Figure 1. GAN can be considered as a *minimax game*:

$$L_{GAN} = L_G = -L_D$$

where L_G and L_D are the cost function of the generator and discriminator respectively. The loss function can be defined as follow using *cross entropy* [11]:

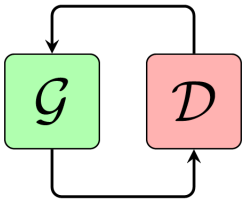

Figure 1: GAN general architecture illustration, inspired by [1]

$$L_{GAN} = E_{x\sim p_r} logD(x) + E_{z\sim p_g} log[1 - D(G(z))] \quad (1)$$

where p_r is the distribution of real data and p_g is the distribution of generated data (the notations are taken from [1]). The discriminator parameters θ^D are then updated using the gradient

$$\nabla_{\theta^D} L_D(\theta^D, \theta^G) = -\nabla_{\theta^D} \frac{1}{N} \sum_{n=1}^{N} [logD(x^n) + log(1 - D(\tilde{x}^n))] \quad (2)$$

where θ^G denotes for generator parameters, samples x are from p_r and $\tilde{x} = G(z)$ are from the generated data distribution $p_g(z; \theta^G)$. The generator G will be updated after by stochastic gradient descent:

$$\nabla_{\theta^G} L_G(\theta^G) = \nabla_{\theta^G} \frac{1}{N} \sum_{n=1}^{N} log(1 - D(\tilde{x}^n)) \quad (3)$$

Theoretically, both generator and discriminator are continuously improved until p_g converges to p_r. One disadvantage of GAN is that there's no accurate representation for the distribution of generated data p_g. Another disadvantage is that if D training and G training processes are not balanced (G is trained too much without updating D), then many generated values of z will be collapsed to the same value of x and G could not be able to model the real data distribution [12].

GANs have shown high effectiveness for image generation [12] and recently have been applied in text generation with promising results [35] [13].

3.5 Embeddings

Before we do the similarity score calculating or the classification, words and documents need to be represented as numeric vectors. Hence, the theory of some embedding methods that we use in our experiments are mentioned here.

3.5.1 ***Word2vec and Doc2vec.*** **Word2Vec** gives a numeric representation for each word in the text corpus. It's able to capture the different relations between words, such as synonyms, antonyms, or analogies (for example: vector("King") - vector("Man") + vector("Woman") = vector("Queen")) [21]. There're two popular Word2vec models: Continuous Bag-of-Words (CBOW) and the Skip-Gram as illustrated in Figure 2. The CBOW model predicts the current word based on the surrounding words (context), and the Skip-gram model predicts context given the current word [21].

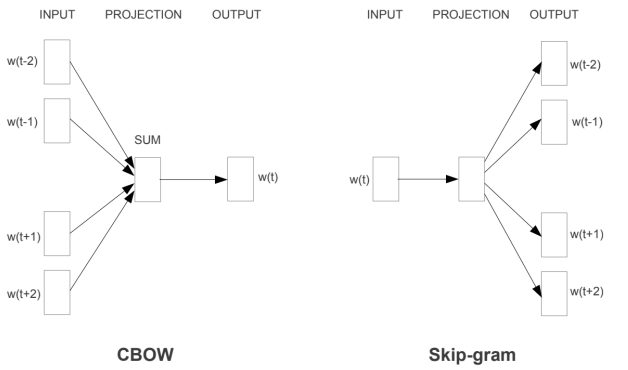

Figure 2: Word2vec new models

Doc2vec, unlike Word2vec, creates representations for documents at any length instead of words. Mikilov and Le [17] have proposed a simple but effective method: they reuse the Word2vec model and add another vector for paragraph ID. When training the word vectors, it also trains the document vector. The model is named as *Distributed Memory version of Paragraph Vector (PV-DM)* and is demonstrated in Figure 3.

3.5.2 Bidirectional Encoder Representations from Transformers (BERT). **BERT** was introduced in late 2018 as a new language representation [7].

Encoder-decoder models are often used in *sequencetosequence* problems such as text summarisation [2]. This model works well with short text but not for long sequences or data with multiple intents. An attention-based mechanism can be applied to deal with this issue. There's a type of deep learning method that built with attention is called a **Transformer** [29] [23].

BERT framework consists of two steps: $pre-training$ and $fine-tuning$. In the pre-training step, the model is trained with unlabeled data. During fine-tuning, the pre-trained parameters are fine-tuned

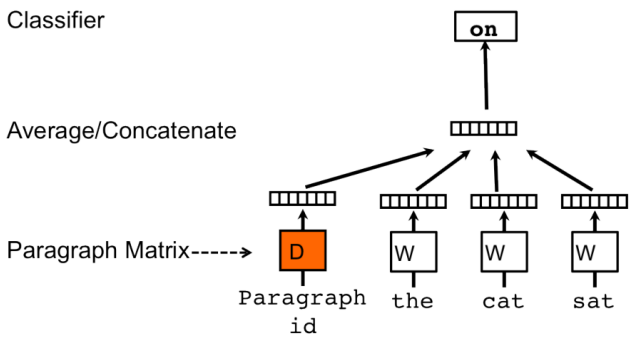

Figure 3: Doc2vec PV-DM model

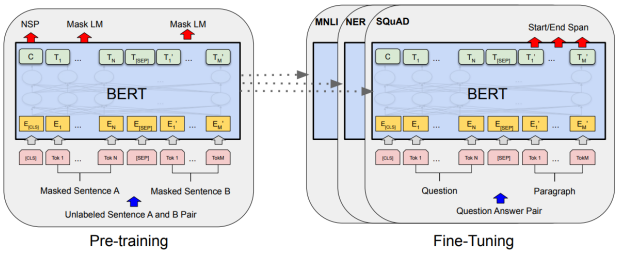

Figure 4: Pre-training and fine-tuning steps in BERT

using labeled data from the downstream tasks [7]. Figure 4 describes those two steps of BERT.

BERT is basically a trained Transformer Encoder stack. Instead of using uni-directional language models for pre-training like the Transformer [23], BERT is fully bi-directional. BERT achieves state-of-the-art performance for multiple NLP tasks, including sentence-level and word-level representations [7]. There're two versions of BERT: base version with an output vector of 728 dimensions for each sequence and large version which outputs a 1024-dimensions vector. We use the large version of BERT in our experiment since it gives a better result when we try with a small part of our dataset.

3.6 BLEU scores

BLEU (Bi-Lingual Evaluation Understudy) is an algorithm to assess the quality of translated or generated text by machine from one natural language to another [32]. Here we use BLEU scores for evaluating the headlines that are generated from the GAN model.

BLEU compares a candidate against multiple references using a *precision* modification. For each n-gram in the candidate, BLEU takes its maximum total count, m_{max}, in any of the references. Those m_{max} are then summed over all distinct n-grams in the candidate and then divided by the total number of n-grams in the candidate to get the modified precision. BLEU's output is always between 0 and 1. The higher the score, the better translation/generation method.

4 DATASET

As mentioned above, there is a number of datasets for fake news related tasks, but they are not directly appropriate to apply for incongruence problem. In this project, we apply the method proposed by [33] to create a dataset from NELA 2017 [14] data. NELA-2017 contains articles from 92 news sources between April 2017 and October 2017. We download the raw text version of this dataset as it is needed for the GAN-based text summarization model to generate the headlines. We also use the preprocessed version of NELA dataset provided by Yoon [33] in order to run their model AHDE as the baseline. Yoon's method creates the synthetic incongruent data with different types to cover various cases the incongruent headlines can happen. There're two rules:

(1) Sampling a number of continuous paragraphs from an article and inserting them into the body of the target article
(2) Sampling a number of discrete paragraphs from one article and inserting them randomly into the body of the target article

Then, the incongruent data is created by applying rule (1) with one or more than one consecutive paragraphs or applying rule (2) with and without random positions to insert.

Further, we still need another dataset to train the body text summarisation model. We choose the CNN / Daily Mail summarization dataset [27] with over 300,000 articles to build the training set and process it into the binary format. Following the instructions from the author of paper [27], we also use Stanford CoreNLP [20] to preprocess and tokenize the data. The Nela-2017 dataset is processed in the same way as CNN / Daily Mail dataset to build the test set for the headline-generating model.

Dataset Statistics:

Table 1 shows the statistical information of dataset used for GAN-based body text summarisation model. Train set and validation set are from CNN / Daily Mail articles and test set is the processed data from NELA-17.

	Number of articles	Percentage
Train	287227	74.7%
Validation	13368	3.5%
Test	84024	21.8%

Table 1: Dataset for headlines generation model

Table 2 describes how we split the dataset for incongruence detecting model. We follow exactly the same split as Yoon [33] proposed in their work to make it easier to compare the result with this baseline. The ratio of congruence and incongruence classes is 1:1 and same for all splits.

	Number of articles	Percentage
Train	71420	85%
Validation	6302	7.5%
Test	6302	7.5%

Table 2: Dataset for incongruent headlines prediction model

The dataset used for prediction models will contain the following fields:

- Original headlines
- Body text
- Generated headlines
- Similarity score between the original headlines and the generated headlines
- Label, 0 is congruent headline and 1 is for incongruent

There're opportunities for other features to add to the dataset, but within the scope of this project, we only use the similarity scores.

5 METHOD AND IMPLEMENTATION

5.1 GAN-based body text summarization model

We apply the implementation of this paper [18] to generate the synthetic headlines by summarizing the body text. The model is similar to the original GAN [12], it includes a generator G and a discriminator D. This implementation is also inspired by SeqGAN [34], one of the early attempts but effective at generating text with GAN. It considers the generator as an agent in reinforcement learning in order to solve the issue of the discrete nature of text.

First of all, both G and D are pre-trained. The generator receives the source text $x = \{w_1, w_2, ..., w_n\}$ and generates the summary $\hat{y} = \{\hat{y}_1, \hat{y}_2, ..., \hat{y}_m\}$. The model uses a bi-LSTM as the encoder and an attention-based LSTM as the decoder. At each time step t, the probability of predicting word from the vocabulary is produced as the following:

$$P_{vocab}(\hat{y}_t) = softmax(V^{'}(V[s_t, c_t] + b) + b^{'}) \tag{4}$$

where $V^{'}, V, b, b^{'}$ are learnable parameters, s_t is the decoder's hidden state and c_t is context vector at time step t. Following the work of See et al. [27], a switching pointer-generator network is used to get the final probability $P(\hat{y}_t)$ of each token in $\hat{y}$.

Once we obtain the pre-trained generator, D is also pre-trained with the negative samples generated from pre-trained G and positive samples from human. The probability of a sequence being original is calculated as:

$$P_{x \sim p_r}(x) = softmax(\delta(f_x)) \tag{5}$$

where δ denotes the max-pooling operation, f_x denotes the multiple features obtained by applying multiple filters with varying window sizes over the input sequence (encoded with a CNN).

During the adversarial process, the generator and discriminator are trained alternatively. The loss function of G includes two parts: $L = \beta L_{DG} + (1 - \beta)L_{ml}$, with L_{DG} is the loss computed by policy gradient as described in section 3.4 and L_{ml} denotes the maximum-likelihood loss, β is the scaling factor to balance the loss function value [18].

5.2 Cosine Similarity Calculation

In order to detect the incongruence problem, our idea is to measure the similarity between the original headline and the headline summarized from the body text instead of computing the similarity between the article's headline and body directly.

The original and generated headlines at first are represented by Doc2vec or Bert. We use the Doc2vec implementation from gensim [24]. For the sentence-level embeddings with BERT, we use the pre-trained model **"bert-large-nli-stsb-mean-tokens"** which is fine-tuned on the AllNLI dataset, then on train set of STS benchmark [25]. This is a model in the large version type of BERT which outputs 1024-dimensions vectors.

5.3 Classification

Once the cosine similarity calculations are done, both the scores and the representation vectors obtained from Doc2vec or BERT are fed to a classifier that we re-implement with modifications from a simple system of multi-layer perceptron (MLP) with one hidden layer. This UCL Machine Reading's (UCLMR) system claimed third place in Stage 1 of the Fake News Challenge 2017 (FNC-1) [26]. The model is illustrated in Figure 5.

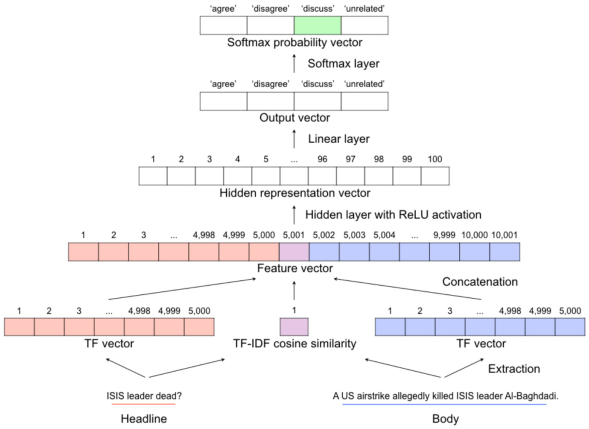

Figure 5: Schematic diagram of UCLMR's system

One modification we make is on the feature vector. Instead of building the feature vector based on term frequency and TF-IDF cosine similarity, the dense input in our model is made from Doc2vec embeddings or BERT embeddings of the original headline and its corresponding generated headline along with the cosine similarity score from the previous step. The length of the feature vector, therefore, is also different: 301 for Doc2vec version and 2049 for BERT version. Another modification is that the length of the output vector is 2 instead of 4 since we only have two classes in the current problem, "congruence" and "incongruence".

We also set up another classification using Support Vector Machine to run the classification using the cosine similarity scores as the unique feature. This actually can also be done with any other simple classifier such as a linear classifier.

6 EXPERIMENTAL EVALUATION

In this section, we report the practical setup of our experiment as well as the results and compare them with the baseline result from Yoon's method [33].

6.1 Experimental setup

Training the GAN model to generate the headlines consumes lots of resources. So we have to borrow one of the GPU servers from the University of Stavanger. The hardware information of this server includes: 2x Xeon E5-2650 v4 @ 2.2GHz (total 48 cores), 264GB RAM and we run the model on the GPU Tesla P100 12GB. For all of the other tasks, we are able to run them on a Macbook Pro with the following configuration: 2.5 GHz Intel Core i7, 16 GB 1600 MHz DDR3, AMD Radeon R9 M370X 2 GB.

We choose the vocabulary size of 50000 for the training process of the headline-generating model. The data for pre-training the discriminator is also available at [18] so we just need to re-use it. The loss function value is initially set to 1e-2, but the model training couldn't get to that point after 1 week so we decide to early stop it at 0.8 for the loss value. Some of other essential parameters:

- dimension of RNN hidden states: 128
- dimension of word embeddings: 128
- minibatch size: 64
- batch size for pre-training discriminator: 64
- max timesteps of encoder (max source text tokens): 400
- max timesteps of decoder (max summary tokens): 100
- learning rate: 0.001
- beam size for beam search decoding: 4

The parameters to initialize Doc2vec:

- distributed memory: dm=1 that means it uses the PV-DM architecture
- ignores all words with total frequency lower than 1: min_count=1
- window size: 10
- dimensionality of the feature vectors: 150
- "noise words" should be drawn for negative sampling: 10
- threshold for configuring which higher-frequency words are randomly downsampled: 1e-4

Since we use the pre-trained BERT model "bert-large-nli-stsb-mean-tokens", there're no other hyperparameters for it. The UCLMR's model [26] is re-implemented using Keras [5] with parameters as below:

- hidden representation vector size: 100
- loss function: *sparse_categorical_crossentropy*
- window size: 10
- optimizer: *adam*
- epochs: 90
- batch size: 500

The detailed instructions to reproduce the experiment can be found it our GitHub repository at https://github.com/khoaln/gan-incongruence-detecting.

6.2 Results

Here we come to the results of both headline-generating model and incongruence detecting model.

For evaluating the quality of generated headlines from the articles' body text, we base on the BLEU scores. The BLEU values of an average of unigram, bigram, trigram, and 4-gram precision are calculated using the method *corpus_bleu* from nltk [19] for each pair of a generated title and the corresponding body. The average BLEU score is about **0.455**. It is relatively similar to the result of

BLEU scores obtained from the experiments of the work of Kai Shu et al. [28] where they generate headlines for clickbait detection.

Table 3 shows the results from our experiments on detecting the incongruence problems. The result of Yoon [33] reproduced with the same dataset is also reported in the table. The comparison is based on accuracy and the area under the ROC curve, AUC.

Model	Accuracy	AUC
Doc2vec + UCLMR	0.536	0.536
BERT + UCLMR	0.601	0.601
Doc2vec + Cosine + SVM	0.566	0.566
BERT + Cosine + SVM	**0.631**	**0.631**
Yoon's model	**0.631**	**0.675**

Table 3: Results of our experiments

We can see from the results that the experiment with cosine similarity scores derived from BERT embeddings for the original and generated headlines and a simple classifier gives the highest result for this project. It is also similar to the result that Yoon's work produces, although that accuracy is much lower than the score they reported in their paper with a Korean dataset.

There are some potential improvements to this approach in the future. We discuss in the next section.

7 FUTURE WORK

While conducting the experiments, we realize that the quality of generated headlines plays an important role in the final result of the detecting model. SeqGAN is known as the foundation of the more advanced attempt, LeakGAN [13] which also gives the highest score in the evaluation results of Texygen [36] (Texygen is a benchmarking platform to support the researches on open-domain text generation models). Therefore, using LeakGAN or another higher effective GAN model could be able to increase the accuracy for incongruence problems detecting model. This could also lead to the need for a new generation method that pays more attention to the attributes of the incongruent headlines when summarizing the body text. Furthermore, due to the lack of large-scale data of incongruent headlines, generating and giving a high-quality dataset to the community will be a valuable contribution for further researches on the topic.

BERT also shows a very high potential in text embeddings as well as in many other NLP tasks. Besides, it's possible to fine-tune the model and train our own embeddings that suits our specific tasks [25]. It's worth to study more on BERT in future work related to incongruence or fake news problems.

8 CONCLUSION

Incongruence between news headlines and real content is a challenging problem and it needs more studies from the scientific community. Our work has presented some specific obstacles in this topic and proposed several simple methods that combine some of the promising technologies such as GAN and BERT to tackle it. Although there are things that need to be improved, we believe that this approach has a bright direction to continue and also expand for other associated problems in the field.

REFERENCES

[1] Johan Björk and Karl Svensson. 2018. Abstractive Document Summarisation using Generative Adversarial Networks. https://hdl.handle.net/20.500.12380/255471

[2] Nikhil Buduma and Nicholas Locascio. 2017. *Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms* (1st ed.). O'Reilly Media, Inc.

[3] Yimin Chen, Niall J. Conroy, and Victoria L. Rubin. 2015. Misleading Online Content: Recognizing Clickbait As "False News". In *Proceedings of the 2015 ACM on Workshop on Multimodal Deception Detection (WMDD '15)*. ACM, New York, NY, USA, 15–19. https://doi.org/10.1145/2823465.2823467

[4] Sophie Chesney, Maria Liakata, Massimo Poesio, and Matthew Purver. 2017. Incongruent Headlines: Yet Another Way to Mislead Your Readers. In *Proceedings of the 2017 EMNLP Workshop: Natural Language Processing meets Journalism*. Association for Computational Linguistics, Copenhagen, Denmark, 56–61. https://doi.org/10.18653/v1/W17-4210

[5] François Chollet et al. 2015. Keras. https://github.com/fchollet/keras.

[6] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014*.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805* (2018).

[8] U. K. Ecker, S. Lewandowsky, E. P. Chang, and R. Pillai. 2014. The effects of subtle misinformation in news headlines. *Journal of experimental psychology: applied* 20, 4 (2014), 323.

[9] William Ferreira and Andreas Vlachos. 2016. Emergent: a novel data-set for stance classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California, 1163–1168. https://doi.org/10.18653/v1/N16-1138

[10] M. Gabielkov, A. Ramachandran, A. Chaintreau, and A. Legout. 2016. Social clicks: What and who gets read on twitter? *ACM SIGMETRICS Performance Evaluation Review* 44, 1 (2016), 179–192.

[11] Ian Goodfellow. 2016. NIPS 2016 Tutorial: Generative Adversarial Networks. (12 2016).

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 2672–2680. http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[13] Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. 2017. Long Text Generation via Adversarial Training with Leaked Information. *arXiv preprint arXiv:1709.08624* (2017).

[14] et al. Horne, Benjamin D. 2018. Sampling the News Producers: A Large News and Feature Data Set for the Study of the Complex Media Landscape. https://github.com/BenjaminDHorne/NELA2017-Dataset-v1

[15] Andrej Karpathy. 2015. The Unreasonable Effectiveness of Recurrent Neural Networks. http://karpathy.github.io/2015/05/21/rnn-effectiveness/

[16] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1746–1751. https://doi.org/10.3115/v1/D14-1181

[17] Quoc Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Eric P. Xing and Tony Jebara (Eds.), Vol. 32. PMLR, Bejing, China, 1188–1196. http://proceedings.mlr.press/v32/le14.html

[18] Linqing Liu, Yao Lu, Min Yang, Qiang Qu, Jia Zhu, and Hongyan Li. 2017. Generative Adversarial Network for Abstractive Text Summarization. (11 2017).

[19] Edward Loper and Steven Bird. 2002. NLTK: The Natural Language Toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1 (ETMTNLP '02)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 63–70. https://doi.org/10.3115/1118108.1118117

[20] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*. 55–60. http://www.aclweb.org/anthology/P/P14/P14-5010

[21] Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. http://arxiv.org/abs/1301.3781

[22] Christopher Olah. 2015. Understanding LSTM Networks. https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[23] Alec Radford and Ilya Sutskever. 2018. Improving Language Understanding by Generative Pre-Training. In *arxiv*. https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_

understanding_paper.pdf
[24] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. http://is.muni.cz/publication/884893/en.
[25] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. http://arxiv.org/abs/1908.10084
[26] Benjamin Riedel, Isabelle Augenstein, Georgios P. Spithourakis, and Sebastian Riedel. 2017. A simple but tough-to-beat baseline for the Fake News Challenge stance detection task. *CoRR* abs/1707.03264 (2017). http://arxiv.org/abs/1707.03264
[27] Abigail See, Peter Liu, and Christopher Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Association for Computational Linguistics*. https://arxiv.org/abs/1704.04368
[28] Kai Shu, Suhang Wang, Thai Le, Dongwon Lee, and Huan Liu. 2018. Deep Headline Generation for Clickbait Detection. https://doi.org/10.1109/ICDM.2018.00062
[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*. Curran Associates Inc., USA, 6000–6010. http://dl.acm.org/citation.cfm?id=3295222.3295349
[30] Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral Multi-Perspective Matching for Natural Language Sentences. (02 2017).
[31] Wikipedia. 2019. Bidirectional recurrent neural networks. https://en.wikipedia.org/wiki/Bidirectional_recurrent_neural_networks
[32] Wikipedia. 2019. BLEU. https://en.wikipedia.org/wiki/BLEU
[33] Seunghyun Yoon, Kunwoo Park, Joongbo Shin, Hongjun Lim, Seungpil Won, Meeyoung Cha, and Kyomin Jung. 2019. Detecting Incongruity between News Headline and Body Text via a Deep Hierarchical Encoder. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 791–800.
[34] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2016. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. arXiv:cs.LG/1609.05473
[35] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI'17)*. AAAI Press, 2852–2858. http://dl.acm.org/citation.cfm?id=3298483.3298649
[36] Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A Benchmarking Platform for Text Generation Models. *SIGIR* (2018).