Abstractive Headline Generation: Statistical versus Deep Learning Approaches

Sujeong Cha, My Phung
New York University Center for Data Science
{sjc433, mtp363}@nyu.edu

Abstract

In the world of digital media, headlines play an important role in guiding readers to find the right content to focus on. Thus, developing an automatic headline generation system that can provide an informative headline from an article becomes a critical challenge. This paper investigates different abstractive text summarization frameworks, including a statistical model introduced in 2000 [Banko et al., 2000] and a Transformer-based model, NHNet, developed in 2020 [Gu et al., 2020] to understand the benefits and limitations of these approaches. Our contributions to this project include 1) successfully implementing and evaluating these models on the recently published dataset, NewSHead, 2) improving the statistical model by adding a pre-trained language model GPT-2, and 3) boosting the performance of NHNet by pre-training its encoder on the Multi-Genre Natural Language Inference (MNLI) dataset.

1 Problem Setting

Nowadays, readers are overwhelmed with the surge of news and information targeting them on different platforms. The numerous numbers of news content and publication platforms including social media have changed the way digital readers consume content. They want to read less but get more of what is important or interesting to them [Verma et al., 2010]. Therefore, headlines have become more important than ever to guide readers towards the right articles or news that they should spend more time on and sometimes serve as the key summary of all they want to know.Therefore, we believe that developing an automatic system that could generate informative and functional headlines from news articles is an important challenge.

A common approach to this challenge is using text summarization and treating headlines as an one-line summary of the document. There are two common approaches in text summarization: extractive and abstractive approaches. Extractive approach aims at selecting top k most important and informative sentences in the document to create a good summary while abstractive approach aims at constructing a summary from scratch using words that appear in the documents. Problems with using extractive methods include 1) the fact that the content of the whole document is rarely included in any single sentence in the document and 2) the length of a typical sentence from the document is larger than an expected length of a headline [Banko et al., 2000]. Therefore, in this paper, we would instead focus on investigating the pros and cons of different abstractive summarization models for the headline generation task. For this purpose, we select two models that were published two decades apart: a statistical model introduced in [Banko et al., 2000] and a deep learning Transformer-based model published in [Gu et al., 2020].

2 Related Work

Headline generation belongs to a broader task of text summarization, in a sense that a headline is usually one short phrase/sentence that captures the overall message while a summary typically spans more than one sentence. As a result, research trends in headline generation closely follow those of text summarization.

One approach to text summarization is an extractive method where a model “extracts” important sentences or paragraphs from the original document. Initial works include [Kupiec et al., 1995], which proposes a binary classifier to select the representative sentences; more recent research focuses on exploiting neural networks via effective representation learning [Cheng et al., 2016]. However, this extractive approach may not be appropriate for headline generation since, by its structural characteristics, it cannot generate summaries shorter than a sentence. According to [Banko et al., 2000], sentences ranked best for summary selection often tend to be even longer than the average sentence in the document. Therefore, in this project, we only focus on abstractive summarization models.

[Banko et al., 2000] suggests an abstractive method that can overcome the aforementioned limitation of extractive summarization. Their model comes from the idea that headline generalization can be treated as a translation problem where a verbose language is translated to a succinct one. Following the structure of statistical machine translation models, their system is composed of 1) a bigram language model that measures how likely a word sequence is, and 2) a conditional summarization model that measures a conditional probability of a word occurring in the summary given that the word appeared in the document. The parameters for both components are estimated by relative frequency based on the training corpus (approximately 25,000 1997 Reuters news-wire articles).

Extending the machine translation inspiration, [Rush et al., 2015] suggests a deep-learning based text summarization. Unlike the previous model which has two separate components, their Attention-Based Summarization (ABS) model directly parametrizes the original distribution as a neural network. This structure closely follows the neural machine translation systems that were an instant hit at the time. Specifically, it utilizes a feed-forward neural language model that contains an attention-based encoder. This encoder is equivalent to the conditional summarization model of the previous noise-channel approach. [Nallapati et al., 2016] further improves on [Rush et al., 2015] by replacing a feed-forward language model with a encoder-decoder GRU-RNN architecture, and also by encoding POS, NER tags, and TF-IDF values along with word embeddings in order to accommodate linguistic characteristics as well. Recent papers that utilizes recent deep-learning development for headline generation include [Hayashi et al., 2018], which uses an LSTM encoder-decoder network, and [Gavrilov et al., 2019] which uses an Universal Transformer architecture paired with byte-pair encoding technique.

Rather than using only word embeddings, [Liu et al., 2019] improves upon Bidirectional Encoder Representations from Transformer [Devlin et al., 2018] model to construct a pre-trained encoder that can create embeddings on a sentence level by adding a [CLS] token at the end of each sentence. More importantly, they introduce a BERT-based document-level encoder, BERTSUM, that captures the semantics of an input document, which could then be used to produce either extractive or abstractive text summarization by varying the top layers. For abstractive summarization, a decoder will be added to the top layers while a classifier will be used to produce extractive summaries.

Continuing the idea of using document-level hidden representations to produce abstractive summaries, [Gu et al., 2020] attempts to perform a single story-headline generation for a set of documents.This task is challenging because a generated headline must capture the information shared by articles but exclude any contents that are specific to each individual article. Moreover, the scarcity of training data further aggravates the problem - while a single article naturally comes with a target variable, namely its headline, expert intervention is needed to annotate a set of documents with a appropriate one-line summary. To address this issue, [Gu et al., 2020] publishes the first manually curated multi-document dataset, NewSHead. It contains 367K news stories (each story has 3-5 articles) published between May 2018 and May 2019. Their work has been a huge inspiration for us to experiment with abstractive summarization based off of their published code and dataset.

3 Approach

3.1 Statistical Approach (Banko)

[Banko et al., 2000] adopt a statistical machine translation approach to tackle headline generation, where they see the verbose document as the source language and the concise version of it – the headline, as the target language. In particular, [Banko et al., 2000] build a probabilistic model that estimate the likelihood of words selected for the headline based on 1) the co-appearance of words in both the document and in the headline, 2) the potential length of the headline, and 3) the likelihood of words following the previously chosen words, which is similar to a language model. This is demonstrated through the following equation:

$$P(w_1, ..., w_n) = \prod_{i=1}^{n} P(w_i \in H | w_i \in D) \cdot P(len(H) = n) \cdot \prod_{i=1}^{n} P(w_i | w_1, ..., w_{i-1}) \tag{1}$$

where w_i denotes the word at position i, H denotes the output headline, and D denotes the input document.

Replication: Implementing from scratch As the model dates back to 20 years ago with no source code available, we implement the model from scratch in Python. The training and decoding process are fairly similar to those of the statistical Part-Of-Speech tagger; we can think of the first term from Eq 1 as the emission matrix, $P(w_i|s_i)$, and the second term as the transition matrix, $P(s_i|s_{i-1})$. Note that [Banko et al., 2000] also follows a first-order Markov assumption, utilizing a bigram language model (i.e. $P(w_i|w_1, \ldots, w_{i-1}) = P(w_i|w_{i-1})$).

One caveat for the training process is that the runtime can act as a great huddle when computing the emission matrix – we have to loop through an entire article for every single vocabulary. The situation might get worse when we train the model on the NewSHead dataset as its articles tend to be lengthier than the Reuters dataset which the original paper trains their model on. Therefore, for the sake of faster training, we 1) encode all texts into Numpy arrays instead of Pandas dataframes, and 2) make use of binary search (`bisect_left` function in Python) when searching through the contents of the articles. Moreover, in order to take care of the unseen words and thereby prevent the division by zero errors, we apply a simple smoothing technique by adding a constant (1e-10) to all terms.

For the decoding process, Viterbi algorithm is employed. One difference with the Viterbi for POS taggers is that we loop through the desired "length" of an input article instead of an input sentence, and loop through each word in the article body instead of POS tags. In addition, we fix the length of the predicted headline to the average headline length in the training set to simplify the decoding process; however, this can be easily extended to variable lengths by changing the input hyperparameter `len` of the implemented Viterbi function.

Further Experiments: Combining with neural LM As an attempt to further improve the statistical model presented by [Banko et al., 2000], we attempt to replace the language model component with a neural language model, Open AI GPT-2, hoping to exploit its pretrained knowledge. (We may call it a "hybrid" model between statistical and neural-based approaches.)

Generative Pre-trained Transformer (GPT) [Radford et al., 2019] is a transformer-based language model trained on millions of web pages called WebText. The architecture of GPT-2 is also large-scaled: 1.5 billion trainable parameters, 1,600 dimensional word embeddings, and 50,257 vocabulary tokens. The main objective of the model is to predict the next word, given all of the previous words, which is similar to the transition matrix in our Eq 1. By the time the paper was published, it achieved state-of-the-art results on 7 out of 8 tested language modeling dataset in zero-shot setting, meaning the model was able to perform well without any training on the target dataset.

It is an undeniable fact that BERT is the go-to model for language pre-training as of now; however, we choose to use GPT-2 as it closely resembles the traditional language models. That is, GPT-2 is autoregressive, capable of predicting the next word given the previous contexts, while BERT is trained to predict the masked tokens conditioned on both left and right contexts.

Replacing the statistical LM with GPT-2 requires an easy fix. Instead of computing a transition matrix during the training phase, we compute it during the decoding stage. While the decoder iterates over the target document (without a headline, of course) in its inner for-loop, we feed the entire input text into the pre-trained GPT-2 model (`GPT2LMHeadModel` in HuggindFace), and get the logit output. Since the shape of the logit output is (batch_size $\times$ input_len $\times$ vocab_size), we take the i^{th} vector in the last dimension where i is the `token_id` we are currently iterating over. For example, if our inner for-loop is at a word "hello" with `token_id` "1000", then we do `logit_output[:, :, 1000]`. This contains individual transition probabilities for the target token conditioned on every word in the document, which we can use as a bigram language model. All other processes of the decoder, including the backward step, remain the same.

3.2 NHNet

The second model we want to experiment with is called NHNet[1], proposed in [Gu et al., 2020], which can produce a single story-headline for a set of relevant news documents. The key advancement of NHNet, compared to other Transformer-based models is its encoder, which creates not only token-level embeddings but also sentence-level and document-level hidden representations for each document.The hidden representations of each single document (single-doc) a in a set A are then combined to output a vector representation of set A. The weights for this linear combination of single-doc embeddings are determined by a self-voting-based document-level attention layer. The encoder is pre-trained as a language model and then jointly pre-trained with the decoder in a single-doc headline generation task. During inference, the model takes in the full set of documents, set A, along with the output sequence produced up to the previous steps (i.e., $[y^1, y^2, ..., y^{i-1}]$), and generates a d_H -dimensional hidden vector representing set A (denoted H^i_A). The end-to-end single-document architecture would predict the next token y^i in the output sequence from H^i_A given the tokens chosen in the previous steps. As mentioned earlier, this model aims at generating a single story headline from multiple news articles while we only aim at generating a title for each single article. This implies that 1) their results are not directly comparable to ours or [Banko et al., 2000], 2) while we preserve the current framework, the interpretation of the model architecture is slightly different, and 3) we need a different pre-training strategy instead of the single-doc headline generation task.

Replication: Model adaptation In terms of model architecture, even though we proceed with one document at a time, we intend to preserve the current framework of the encoder because we hypothesize that training the model to have a good representation of the entire document may benefit the downstream summarization task. Since the input is one document at a time, the self-voting-based document-level attention layer will have no effect on the final hidden representation of the set. During inference, instead of creating a hidden representation of the entire set of document H^i_A, the model takes one document a as an input sequence along with the headline output sequence produced up to the previous step (i.e., $[y^1, y^2, ..., y^{i-1}]$), and generates a d_H -dimensional hidden vector representing document a (denoted H^i_a). The decoder would predict the next token y^i in the predicted title from H^i_a given the tokens chosen in the previous steps (Eq 2). Ultimately the top output sequence is chosen using a beam search to maximize the likelihood of having y^i as the next token given the previous y^1 to y^{i-1}.

$$P(y^i = y|[y^1, ..., y^{i-1}]) = \frac{\exp(H^i_a \cdot M_y)}{\sum_{y' \in V} \exp(H^i_a \cdot M_y)} \tag{2}$$

where V is the entire vocabulary, M_y is the column of a learnable embedding matrix $M \in R^{d_H |V|}$, in correspondence to token y [Gu et al., 2020].

Further Experiments: Pre-training NHNet's encoder on GLUE tasks Speaking of transfer learning strategies, we hypothesize that pre-training NHNet's encoder on a different task instead of headline generation may enhance the capability of the model in understanding natural language and increase its generalizability. Because the NHNet model has a Transformer-based encoder that is initialized with the pre-trained "bert-base-uncased", we experiment with pre-training the encoder on different GLUE tasks. Our inspiration for this pre-training strategy comes from the encouraging results from [Phang et al., 2019] and [Pruksachatkun et al., 2020], which show that pre-training BERT on a natural language understanding task greatly boosts the model performance on downstream tasks. In particular, [Phang et al., 2019] experiments with four different GLUE tasks for pre-training BERT as a sentence encoder, including MNLI, QQP, Real/Fake, and SNLI. Among which, pre-training on a Multi-Genre Natural Language Inference corpus, which is referred to as MNLI [Williams et al., 2018], consisting of 433K cross-genre sentence pairs annotated with textual entailment information, seems to be the most beneficial as it increases the results on four out of eight GLUE tasks (excluding MNLI) during evaluation. This result motivates us to include MNLI as one of our pre-training experiments.

The second natural language understanding task/dataset we want to explore is the Microsoft Research Paraphrase Corpus (MRPC) introduced in [Dolan et al., 2005], which contains 5,801 sentence pairs collected from newswire articles and each pair is labelled if it is a paraphrase or not by human annotators. We believe that pre-training our encoder on this task will benefit our target task, headline

[1]https://github.com/tensorflow/models/tree/master/official/nlp/nhnet

generation, for two reasons. First, the example sentences from MRPC are extracted from news articles, which may share the same semantics as our target dataset. Second, MRPC requires the model to classify if two sentences capture a paraphrase/semantic equivalence relationship or not, which requires complex reasoning and inference at a sentence level. This level of deep understanding of natural language is directly relevant and beneficial to our summarization task.

Last but not least, we experiment with sequential pre-training which is mentioned in [Pfeiffer et al., 2021]. According to [Pfeiffer et al., 2021], sequential pre-training implies that we first pre-train our model on task I, namely MNLI, then further pre-train it on task II, namely MRPC. The paper claims that this method may lead to a degradation in the target task due to catastrophic forgetting, meaning that the model may have forgotten what it has learned in task I when it is finetuned on the third task, our target task. Since we have not found a specific experiment result that explicitly demonstrates the effect of catastrophic forgetting, we want to include this in our set of experiments.

4 Dataset

For our experiments, we choose the NewSHead dataset published by [Gu et al., 2020] because, to the best of our knowledge, it is the largest news article dataset containing more than 0.9M documents. Furthermore, it is a fairly recent dataset published between May 2018 and May 2019, compared to those dataset that are commonly used in text summarization research (Gigaword released in 2003; DUC released in 2003 and 2004).

One thing to reiterate is that this dataset is developed for multi-document summarization while our project focuses on headline generation for a single article. The original dataset is composed of 359,940 “stories” for training, 5,000 for validation, and 5,000 for testing where each story contains three to five related articles. The headline for each story is manually generated by crowd-sourced curators and serves as a target variable. Considering this difference in objective, we only take the first article from each story group and use them as our single document inputs. We experiment with two types of labels: 1) the original title of each article and 2) the story headline. Due to limited computing resources, we only crawl and preprocess the urls given in the original validation and test sets and repartition a total of nearly 10,000 news articles into our training, validation, test sets using the 8:1:1 ratio.

5 Results

5.1 Evaluation Metrics

Evaluating the performance of a text summarization model is a challenging task itself which has been admitted by [Banko et al., 2000] and [Gu et al., 2020]. Therefore, it is more reliable to develop a system of evaluation metrics instead of one to obtain more accurate judgements of our selected models. To this goal, we use the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) metrics introduced in [Lin, 2004], particularly its two variants, ROUGE-1 F and ROUGE-L F. ROUGE-1 measures the similarity between the ground truth and the candidate headlines, by computing the percentage of words in the ground truth that appears in the predicted headline. One key parameter in specifying our ROUGE metrics is n, the number of n-grams we use to make our comparison, denoted ROUGE-N. In our experiments, we choose ROUGE-1 (unigram) over ROUGE-N because headline summary is relative short and unigram comparison is sufficient. Furthermore, inspired by the evaluation system in [Gu et al., 2020], we use ROUGE-F, an F-1-oriented version of ROUGE computed as in (Eq 3, 4, 5). Combining unigram and F1 frameworks, our first evaluation metrics is ROUGE-1 F.

$$ROUGE_{1-R} = \frac{\sum_{S \in PredictedHeadline} \sum_{unigram \in S} count_{match}(unigram)}{\sum_{S \in GroundTruthHeadline} \sum_{unigram \in S} count(unigram)} \tag{3}$$

$$ROUGE_{1-P} = \frac{\sum_{S \in PredictedHeadline} \sum_{unigram \in S} count_{match}(unigram)}{\sum_{S \in PredictedHeadline} \sum_{unigram \in_{1-P} S} count(unigram)} \tag{4}$$

$$ROUGE_{1-F} = 2 \cdot \frac{ROUGE_{1-R} \cdot ROUGE_{1-P}}{ROUGE_{1-R} + ROUGE_{1-P}} \tag{5}$$

Moreover, another critical variant of the ROUGE metrics is ROUGE-L, which measures the longest common sequence between the ground truth and generated headline. One benefit of using ROUGE-L is that it awards credit to in-sequence unigram matches, which allows us to compare the structure of the two sentences. For example, given the ground truth sentence, "Police killed the gunman." (S1) and two candidates, "Police killed the gunman." (S2) and "The gunman killed police." (S3), we would obtain the same ROUGE-1 score for both sentences (which is 1.00). However, considering the order of the words, ROUGE-L gives a higher score to S2 (1.00) as the words are in the same order as in S1 while giving S3 half of the score (0.5) since the longest common sequence between S1 and S3 only has two words, "The gunman" [Lin, 2004]. By selecting F1 as a base for computation, our second evaluation metrics is ROUGE-L F, which is computed similarly to ROUGE-1 F, except for switching from the count of overlapping words to the longest common subsequence.

5.2 Banko

Replication Results Implementing the statistical model as explained in 3.1., we achieve comparable performance to the original paper. By the time the paper was published, typical metrics such as ROUGE were not available, so the authors utilize their own way of measure called "Word Overlap" where they record 1) the maximum overlap between the actual headline and the generated headline, and 2) the length at which that maximum overlap is observed (Results in Appendix A). We believe this word overlap metric is most similar to the ROUGE-1 Recall. Comparing based on these two metrics, our replication achieves a marginally higher score (Table 1; 0.1754 vs. 0.2220). However, this difference is expected as the paper does not give detailed explanation on data preprocessing and train/valid/test set splitting and therefore we had to exercise our own discretion. Considering these practical issues, our model can be considered as a successful replication of [Banko et al., 2000] despite the marginal gap in performance.

Table 1: Results on Test Set: Banko Replication on Reuters and NewSHead

Evaluation Metrics	Reuters	NewSHead
Word overlap (Banko et al., 2000)	**0.1754**	-
ROUGE-1 R (our replication)	**0.2220**	0.1085
ROUGE-1 F (our replication)	0.2459	0.1354
ROUGE-L F (our replication)	0.2581	0.1345

Table 1 also tells us that the performance on NewSHead is almost 50% worse than the performance on Reuters. This degradation might come from the fact that the NewSHead dataset contains wider topics with a wider vocabulary set while Reuters is primarily focused on the financial market - in fact, Reuters has 63,375 unique tokens while NewSHead has 88,299. Another possible explanation is that the actual headlines of NewSHead tend to be longer and thus harder to predict. The maximum headline for NewSHead is 38 words while it is only 11 words for Reuters.

Banko with GPT-2 Our attempt to enhance the original statistical model by replacing the language model with a neural LM shows a small but meaningful improvement. When we use the vanilla version of pre-trained GPT (`GPT2LMHeadModel.from_pretrained('gpt2')`), its ROUGE-1 F score improves by 0.03; it seems like a marginal improvement in terms of number, but the generated predictions are much more "readable". For example, one headline generated by the original replication is `''richter creek wildfire near osoyoos with the''`, while the one produced by the modified model sounds more fluent: `''wildfire at the Richter Creek fire''`.

In order to further improve the performance, we experiment with fine-tuning the GPT-2 with our NewSHead dataset. However, when fine-tuned with the training set, the performance rarely improves: from 0.1644 ROUGE-1 F to 0.1669. One plausible hypothesis is that the training set does not help simply because it is very different from the test set, containing dissimilar contents. To test this hypothesis, we fine-tune the GPT-2 with the test set instead of the train set. (This is surely not a data leakage because we do not include the headlines from the test set when fine-tuning.) Compared to the baseline, this model fine-tuned with the test set reaches a ROUGE-1 F of 0.1805, equivalent to 33% improvement.

Table 2: Results on NewSHead's Test Set: Banko with GPT-2 as neural LM

Model	ROUGE-1 F	ROUGE-L F
Banko et al. (Baseline)	0.1354	0.1345
Banko w/ Pre-trained GPT-2	0.1644	0.1676
Banko w/ Fine-tuned (on training set) GPT-2	0.1669	0.1604
Banko w/ Fine-tuned (on test set) GPT-2	**0.1805**	**0.1726**

5.3 NHNet

Impact of Transfer Learning We first replicate the original NHNet model on our single-doc dataset using articles' original title as the labels. We use "bert-base-uncased" as our initial checkpoint for the encoder and train the model with the default setup as shown in the NHNet's paper and git repo (Adam optimizer with a scheduler and an initial learning rate of 0.05) except for the number of training epochs. We have to limit the number of training epochs due to lack of high performance computing resources. The result from this replication is considered as our baseline.

In the second experiment, we add a classification layer on top of BERT and pre-train the entire model on the MNLI dataset for 3 epochs, using a standard learning rate of 2e-5 and Adam optimizer with weight decay (AdamW). After this pre-training round, the weights of the BERT encoder will be used to initialize the NHNet's encoder. We then train the entire NHNet model on our target task, headline generation on NewSHead data, and call this model NHNet with MNLI-pretrained encoder. In the third experiment, namely NHNet with MRPC-pretrained encoder, we follow the same procedure and training setup as our second experiment, using MRPC instead of MNLI as our pre-training task. Lastly, to experiment with sequential pre-training, we use the MNLI-pretrained encoder as our initialization, add a classification layer on top, and train the whole model on the MRPC dataset. The MNLI-MRPC pretrained encoder will then be trained together with NHNet on our headline generation task. The result is reported in Table 3.

Table 3: Results on NewSHead's Test Set: NHNet with Pretrained Encoder

Model	ROUGE-1 F	ROUGE-L F
NHNet (Baseline)	0.1412	0.0701
NHNet w/ MNLI-pretrained Encoder	**0.1823**	**0.1109**
NHNet w/ MRPC-pretrained Encoder	0.1519	0.0825
NHNet w/ MNLI-MRPC-pretrained Encoder	0.1603	0.0915

The results show that transfer learning significantly improves model performance, especially when we choose the right task and dataset for pre-training. Pre-training the encoder on MNLI task and dataset improves ROUGE-1 F by 29.1% and ROUGE-L F by 58.2%. Using MRPC for pre-training also boosts NHNet's performance but not as significantly. One possible explanation is that compared to MRPC, MNLI is a much larger dataset (about 74 times the size) and covers a wider variety of domains and linguistic styles, not just news documents. Last but not least, it is interesting to observe that if we continue pre-training the encoder on the second task, MRPC, after pre-training it on MNLI, NHNet's performance degrades. This result confirms the claim mentioned in [Pfeiffer et al., 2021] that sequential pre-training may not improve model performance at the target task due to cause catastrophic forgetting.

Compared to the result in [Gu et al., 2020], our results are not as high for three reasons. First, due to the limited computing resources, we can only train our model for 10-20 epochs (which is equivalent to 10-20 hours of GPU) whereas the default of NHNet is 100 epochs (estimated 100 hours of GPU). We do see that the longer the training time is, the better result we get (Appendix B). Second, the original dataset contains 359,940 training examples where we only use 8,000 due to the same reason as above. Last but not least, our result is not directly comparable to the paper. It is because their label is story headlines, which are human-annotated, short, and fact-based while our label is original titles of the articles, which are subject to the author's writing style, intended to be catchy, and may contain complex expressions (ie. idioms, metaphors). Therefore, generating an article's title is expected to be more challenging. This hypothesis is validated in our second set of experiments.

Story Headline versus Original Title Label Another interesting set of experiments we have done is varying the ground truth labels, the original title of an article or a human-annotated story, which could be interpreted as the key point or topic of the article. We hypothesize that generating headlines that are close to the original titles is more challenging than predicting the story headlines. It is because article titles are intended to catch people's attention and are subjective to the authors' writing style. With that being said, original titles may contain intentionally exaggerated keywords, idioms, metaphors, or generally complicated expressions that may not necessarily share the same set of words as the articles themselves. In contrast, story labels tend to be shorter, to-the-point, and event-based, which are more likely to share the same set of words as the articles. In this experiment, we focus on two models, the baseline and NHNet with MNLI-pretrained encoder - our best model found in the previous experiment and the results are reported in Table 4.

Table 4: Results on NewSHead's Test Set: NHNet Trained with Story vs. Title Label

	ROUGE-1 F		**ROUGE-L F**	
Model	Story Headline	Original Title	Story Headline	Original Title
NHNet	**0.2455**	0.1412	**0.0910**	0.0701
W/ MNLI	0.2319	**0.1823**	0.0888	**0.1109**

The results confirm our hypothesis that generating the original title of a news article is a much more challenging task compared to generating a story headline. One interesting observation is that NHNet with MNLI-pretrained encoder does not outperform the original model when the label is story headlines. The trend is entirely opposite when the label is the original title. This could be because predicting story headlines is an easier task than predicting titles; thus, pre-training the encoder in this case is not necessary and could even slow down the model's adaptation on the target dataset. From this, we can conclude that transfer learning has a larger positive impact when the target task is more difficult.

5.4 Statistical versus Deep Learning Models

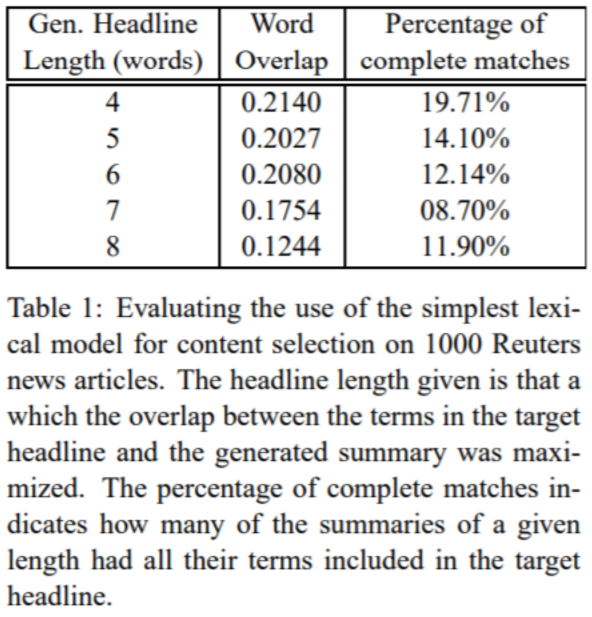
Comparing the best performing experiment from each model (Banko: 0.1805 ROUGE-1 F, 0.1726 ROUGE-L F / NHNet: 0.1823 ROUGE-1 F, 0.1109 ROUGE-L F), one possible conclusion we can draw is that the statistical model can achieve a comparable result to that of the deep-learning based model. However, we should keep in mind that NHNet is not ran on its full-blown capacity; due to limited computing resources, we have to limit the number of epochs tried to 30 epochs at max, and the volume of training set is cut to 2.2% of the original data for the same reason. Rather, it would be more appropriate to state that *given limited data and computing power*, a statistical model can still equally perform well. That is, a traditional statistical model can still play an important role in this era where only deep-learning models are valued and sought after.

6 Conclusion

In this project, we tried to tackle the abstractive headline generation task, which still remains as a challenging problem in the NLP domain. The major outcomes of this project include 1) a from-scratch implementation of an old statistical model, and 2) a replication of a complicated deep-learning based model. In addition, we successfully improved the performance of the statistical and deep-learning model, respectively by 1) replacing the frequency-based LM with a neural LM, and 2) pre-training the model's encoder with different-task but higher-volume datasets.

Going through numerous challenges during experiments, we could come across the quandaries that researchers are currently facing - even the most recent models, NHNet, for example, report a ROUGE-1 score around 0.6 (let alone the fact that ROUGE metrics themselves have many pitfalls). In addition, based on our experience, building an abstractive headline generation model that produces a title that sounds as natural as a human writer is still likely to require decades of research. For instance, many of the generated headlines from our "best" models did not sound fluent even though we incorporated language models. Although our experiments had to be restricted to a fewer epoch and a smaller dataset, it would be interesting to carry out future experiments where no such limitation is imposed and compare with our current figures.

Appendix A. Performance of [Banko et al., 2000]



Gen. Headline Length (words)	Word Overlap	Percentage of complete matches
4	0.2140	19.71%
5	0.2027	14.10%
6	0.2080	12.14%
7	0.1754	08.70%
8	0.1244	11.90%

Table 1: Evaluating the use of the simplest lexical model for content selection on 1000 Reuters news articles. The headline length given is that a which the overlap between the terms in the target headline and the generated summary was maximized. The percentage of complete matches indicates how many of the summaries of a given length had all their terms included in the target headline.

Appendix B. Performance of NHNet When Increasing Training Epochs

Number of Training Epochs	ROUGE 1-F	ROUGE L-F
10	0.1357	0.0701
20	0.1412	0.0701
30	0.1686	0.0948

References

[Verma et al., 2010] Verma, Jyoti, and Vijetacharya Malviya. “The Impact of Internet and Digital Media on Reading Habit.” XXIV National Seminar of the IASLIC, vol. 50, Dec. 2010.

[Banko et al., 2000] Banko, Michele, et al. “Headline Generation Based on Statistical Translation.” Proceedings of the 38th Annual Meeting on Association for Computational Linguistics - ACL ’00, 2000, doi:10.3115/1075218.1075259.

[Kupiec et al., 1995] Kupiec, Julian, et al. “A Trainable Document Summarizer.” Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR ’95, 1995, doi:10.1145/215206.215333.

[Cheng et al., 2016] Cheng, Jianpeng, and Mirella Lapata. “Neural Summarization by Extracting Sentences and Words.” Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2016, doi:10.18653/v1/p16-1046.

[Rush et al., 2015] Rush, Alexander M., et al. “A Neural Attention Model for Abstractive Sentence Summarization.” Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2015, doi:10.18653/v1/d15-1044.

[Nallapati et al., 2016] Nallapati, Ramesh, et al. “Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond.” Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning, 2016, doi:10.18653/v1/k16-1028.

[Liu et al., 2019] Liu, Yang, and Mirella Lapata. “Text Summarization with Pretrained Encoders.” Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019, doi:10.18653/v1/d19-1387.

[Devlin et al., 2018] Devlin, Jacob, et al. “BERT: Pre-Training of Deep Bidirectional Transformers for Language.” CoRR, vol. 1810.04805, 2018.

[Hayashi et al., 2018] Hayashi, Yuko, and Hidekazu Yanagimoto. “Headline Generation with Recurrent Neural Network.” New Trends in E-Service and Smart Computing, 2018, pp. 81–96., doi:10.1007/978-3-319-70636-8_6.

[Gavrilov et al., 2019] Gavrilov, Daniil, et al. “Self-Attentive Model for Headline Generation.” Lecture Notes in Computer Science, 2019, pp. 87–93., doi:10.1007/978-3-030-15719-7_11.

[Gu et al., 2020] Gu, Xiaotao, et al. “Generating Representative Headlines for News Stories.” Proceedings of The Web Conference 2020, 2020, doi:10.1145/3366423.3380247.

[Lin, 2004] Lin, Chin-Yew. “ROUGE: A Package for Automatic Evaluation of Summaries.” Association for Computational Linguistics, Text Summarization Branches Out, 2004.

[Radford et al., 2019] Radford, Alec, et al. "Language Models are Unsupervised Multitask Learners.", OpenAI Blog, 1(8), 2019

[Phang et al., 2019] Phang, Jason, et al. "Sentence Encoders on STILTs: Supplementary Training on Intermediate Labeled-data Tasks.", arXiv:1811.01088 [cs.CL]

[Pruksachatkun et al., 2020] Pruksachatkun, Yada, et al. "Intermediate-Task Transfer Learning with Pretrained Models for Natural Language Understanding: When and Why Does It Work?", arXiv:2005.00628 [cs.CL]

[Williams et al., 2018] Williams, Adina, et al. "A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference.", arXiv:1704.05426v4 [cs.CL]

[Dolan et al., 2005] Dolan, William, et al. "Automatically Constructing a Corpus of Sentential Paraphrases.", Third International Workshop on Paraphrasing (IWP), 2005

[Pfeiffer et al., 2021] Pfeiffer, Jonas, et al. "AdapterFusion: Non-Destructive Task Composition for Transfer Learning.", arXiv:2005.00247v3 [cs.CL]