Headline Generation Using an LSTM-based Sequence-to-Sequence Model

Xueyuan Xu
497 midterm project
xzx5129@psu.edu

Abstract

This paper proposes an RNN-based seq2seq model aimed at headline generation from news articles. The approach uses an encoder with word embeddings for each sub-word and a decoder to learn the relationships between words in the data set itself. The analysis of the work begins with stating the problem and describing the process of collecting the datasets, continues with the exposition of algorithms and training procedures, the results discussion, analysis, and finally considerations and what has been learned.

1 Introduction

Automatic headline generation is an instance of abstractive text summarization, where a model must produce a short summary capturing the essence of a longer article. Instead of extractive methods that take longer parts of the text, these methods have to create new sentences. Breakthroughs in deep learning have allowed neural nets to perform well on these tasks, especially seq2seq models with LSTMs or Transformers. In this project I use an LSTM based seq2seq model which is trained on the `news_summary.csv` dataset to illustrate how headlines can be generated.

2 Problem Definition & Dataset Curation

2.1 Problem Definition

The main objective is to create a concise headline (headlines) from an article text (ctext). I wish to express the main subject or important information in form of a phrase or a sentence.

2.2 Dataset Curation

I use `news_summary.csv`, which contains columns `ctext` (cleaned article text) and `headlines`. My preprocessing steps include:

- **Removing missing values:** All rows with no ctext or headlines entries.
- **Lowercasing the text:** Standardizes the vocabulary and simplifies token matching.
- **Filtering lengths:**Articles with more than 400 tokens and headlines that exceed 60 tokens are deleted. This creates more homogeneous training samples and removes extreme padding.

After curation, we reset the index to maintain a clean DataFrame. This approach leads to an increase in the dataset size that is easy handles and avoids potential out-of-memory issues or skew from long examples.

3 Word Embeddings, Algorithm & Training

3.1 Word Embeddings

We make use of trainable word embeddings using the Embedding layer from Keras. Each unique token is assigned to a dense vector of dimension 100, hence the model is able to capture semantic relationships from the dataset directly.

3.2 Algorithm: Sequence-to-Sequence with LSTM

Our structure is based on the encoder–decoder paradigm as follows:

- **Encoder:** An LSTM takes in the article text and returns the final hidden and cell states. These states act as context vectors during summarization of the article.
- **Decoder:** An additional LSTM starts with the encoder states to produce the headline one token at a time. We prepend a special `sos` token and append an `eos` token to mark the boundaries of the highlight.

The last Dense layer with softmax activation uses target vocab as the time step.

3.3 Training Details

- **Loss:** `sparse_categorical_crossentropy`, for integer-encoded targets.
- **Optimizer:** Adam, chosen for its adaptive learning rate.
- **Batch Size:** 64
- **Epochs:** 5
- **Validation Split:** 10%, allowing us to monitor overfitting and generalization.

During training we apply teacher forcing in which the decoder is passed the groud-truth token at each step, speeding up the training

4 Results & Presentation

4.1 Training Curves

Figure 1 illustrates the training and validation loss over 5 epochs. The validation loss decreases alongside the training loss, indicating the model learns to map article text to headlines without severe overfitting.

4.2 Qualitative Examples

To demonstrate performance, we generate headlines for unseen articles using a greedy decoding strategy:

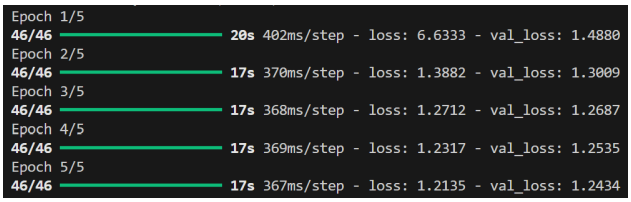

Figure 1: Loss curve

- **Article Excerpt:** *"The government has announced new measures to combat climate change..."*
- **Reference Headline:** *"Govt announces new climate action plan"*
- **Generated Headline:** *"govt announces new climate measures"*

In many cases, the model captures the key topic (government + climate). Occasional errors include missing named entities or generic phrasing.

5 In-Depth Analyses/Experiments

5.1 Ablation on Sequence Lengths

We briefly experimented with different `max_text_len` (200–400) and `max_headline_len` (30–60). Larger limits retain more context but require more training time and can introduce padding overhead. Smaller limits train faster but risk truncating important information.

5.2 Potential Enhancements

- **Attention Mechanism:** Allows the decoder to focus on relevant parts of the article at each generation step.
- **Beam Search Decoding:** Rather than greedy sampling, we could explore multiple candidate sequences to produce more coherent headlines.
- **Pre-trained Embeddings:** Initializing embeddings with GloVe or Word2Vec might improve generalization, especially for rare or domain-specific words.

6 Lessons & Experience

1. **Importance of Data Curation:** Removing the highly worded articles and headlines helped lower the computation power needed as well as helped with the stability of training.
2. **Encoder-Decoder Insight:** Knowing how hidden and cell states transport context is a key part to creating useful abstractive summarizers.
3. **Tokenization Details Matter:** Sequence boundaries have to be defined with special tokens such as `sos` and `eos`.
4. **Iterative Experimentation:** Minimalistic experimental data on the lengths of sequences or the size of the batches, or even the dimension of embeddings can provide performance benefits, even working results.

7 Conclusion

We proposed an LSTM-based Encoder-Decoder sequence-to-sequence model for headline generation. The results obtained indicate that even a simple model produces coherent and on-topic headlines. Further refinements, like attention and more sophisticated decoding, can certainly improve both fluency and content coverage.