A Hierarchical Model for Text Autosummarization

Hrishikesh Vaidya, Tapan Sahni, Akilesh B
Indian Insitute of Technology, Hyderabad
{cs13b1035,cs13b1030,cs13b1042}@iith.ac.in

1 Introduction

Summarization is a central task in natural language processing. Most summarization systems are *extractive* methods. However, *extractive* methods are limited by their nature, as summaries do not essentially come from the source. On the other hand, *abstractive* methods generate summaries from the source, although they may not appear as part of the original text.

Recent LSTM models [2] have shown powerful results on generating meaningful and grammatical sentences in sequence generation tasks like machine translation [4] [1], [3] or parsing [5]. This performance is at least partially attributable to the ability of these systems to capture local compositionally: the way neighboring words are combined semantically and syntactically to form meanings that they wish to express.

In this project, an encoder-decoder model is used to summarize a news article into its title. We develop hierarchical LSTM models that arranges tokens, sentences and paragraphs in a hierarchical structure, with different levels of LSTMs capturing compositionality at the token-token and sentence-to-sentence levels. More specifically, a hierarchical LSTM model is used as an encoder, in which the representations of sentences are learned by a LSTM model whose inputs are words, and the representation of the document is learned by another LSTM model whose inputs are sentences representations. A normal LSTM is used as a decoder.

2 Motivation

Effective text summarization has the potential to allow people to process more information. This is generally relevant to longer document as their intent is difficult to process. Recent LSTM models have shown significantly better results to capture the semantics of sentences. Could these models be extended to deal with generation of larger structures like paragraphs or even entire documents? In standard sequence to sequence generation tasks, an input sequence is mapped to a vector embedding that represents the sequence, and then to an output string of word. Multi-text generation tasks like summarization works in a similar way. Just as the local semantics of words can be captured by LSTM models, can the semantics of higher-level text units (e.g., clauses, sentences, paragraphs, and documents) be captured in a similar way.

3 Long Short Term Memory(LSTM)

In this section we give a brief overview of LSTMs as described in [2]. LSTMs are a special kind of recurrent networks capable of learning long term dependencies. In LSTM each cell state is associated with an input gate, output gate, forget gate and memory gate denoted as i_t, o_t, f_t, c_t. The gating mechanism is used to determine the extent of input information being stored in the cell state. For input sequence $X = \{x_1, x_2, x_3 \dots x_{N_x}\}$, let e_t denote the input embedding at time step t while h_t denote the hidden state vector computed using e_t and h_{t-1}. σ is the sigmoid activation function. The vector representaton of hidden vector h_t is given by:

$$\begin{bmatrix} i_t \\ f_t \\ o_t \\ l_t \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{bmatrix} W \begin{bmatrix} h_{t-1} \\ e_t \end{bmatrix}$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot l_t$$

$$h_t = o_t \cdot c_t$$

where $W \in \mathbf{R}^{4K \times 2K}$. In sequence-to-sequence generation tasks, an input sequence is paired with an output sequence $Y = \{y_1, y_2, y_3, \dots y_{n_Y}\}$. An LSTM learns a distribution over outputs and sequentially predicts tokens using a softmax function:

$$\begin{aligned} P(Y|X) &= \prod_{t \in [1, n_Y]} p(y_t | x_1, x_2, \dots, x_t, y_1, y_2, \dots, y_{t-1}) \\ &= \prod_{t \in [1, n_y]} \frac{\exp(f(h_{t-1}, e_t)}{\sum_{\prime y} \exp(f(h_{t-1}, e_{\prime y}))} \end{aligned}$$

$f(h_{t-1}, e_{y_t})$ denotes the activation function between h_{t-1} and e_{y_t}, where h_{t-1} is the representation obtained from the LSTM at time $t-1$.

4 Methodology

4.1 Model 1: Standard LSTM

The whole input and output are treated as one sequence of tokens. Following Sutskever et al. (2014) and Bahdanau et al. (2014), we trained an autoencoder that first maps input documents into vector representations from a $LSTM_{encode}$ and then reconstructs inputs by predicting tokens within the

document sequentially from a $LSTM_{decode}$. Two separate LSTMs are implemented for encoding and decoding with no sentence structures considered.

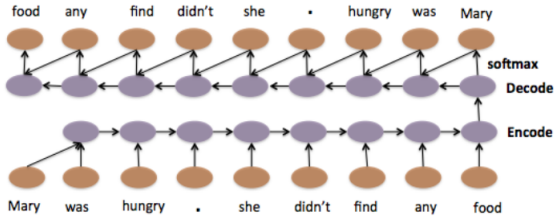

Figure 1: Standard Sequence to Sequence Model.

4.2 Model 2 : Hierarchical LSTM

The hierarchical model draws on the intuition that just as the juxtaposition of words creates a joint meaning of a sentence, the juxtaposition of sentences also creates a joint meaning of a paragraph or a document.

Encoder We first obtain representation vectors at the sentence level by putting one layer of LSTM (denoted as $LSTM^{word}_{encode}$) on top of its containing words:

$$h^w_t(enc) = LSTM^{word}_{encode}(e^w_t, h^w_{t-1}(enc)) \quad (1)$$

The vector output at the ending time-step is used to represent the entire sentence as $e_s = h^w_{end_s}$

To build representation e_D for the current document/paragraph D, another layer of LSTM (denoted as $LSTM^{sentence}_{encode}$) is placed on top of all sentences, computing representations sequentially for each timestep:

$$h^s_t(enc) = LSTM^{sentence}_{encode}(e^s_t, h^s_{t-1}(enc)) \quad (2)$$

Representation $e^s_{end_D}$ computed at the final time step is used to represent the entire document: $e_D = h^s_{end_D}$

Thus one LSTM operates at the token level, leading to the acquisition of sentence-level representations that are then used as inputs into the second LSTM that acquires document-level representations, in a hierarchical structure.

Decoder As with encoding, the decoding algorithm operates on a hierarchical structure with two layers of LSTMs. LSTM outputs at sentence level for time step t are obtained by:

$$h^s_t(dec) = LSTM^{sentence}_{decode}(e^s_t, h^s_{t-1}(dec)) \quad (3)$$

The initial time step $h^s_0(d) = e_D$, the end-to-end output from the encoding procedure, $h^s_t(d)$ is used as the original input into the $LSTM^{word}_{decode}$ for subsequently predicting tokens within sentence $t+1$. $LSTM^{word}_{decode}$ predicts tokens at each position sequentially, the embedding of which is then combined with earlier hidden vectors for the next time step prediction until the end_s token is predicted. The procedure can be summarized as follows:

$$h^w_t(dec) = LSTM^{sentence}_{decode}(e^w_t, h^w_{t-1}(dec)) \quad (4)$$

$$p(w|.) = softmax(e_w, h^w_{t-1}(dec)) \quad (5)$$

During decoding, $LSTM^{word}_{decode}$ generates each word token w sequentially and combines it with earlier LSTM-outputted hidden vectors. The LSTM hidden vector computed at the final time step is used to represent the current sentence.

This is passed to $LSTM^{sentence}_{decode}$, combined with h^s_t for the acquisition of h_{t+1}, and outputted to the next time step in sentence decoding.

For each timestep t, $LSTM^{sentence}_{decode}$ decide whether decoding should proceed or come to a full stop: we add an additional token end_D to the vocabulary. Decoding terminates when token end_D is predicted.

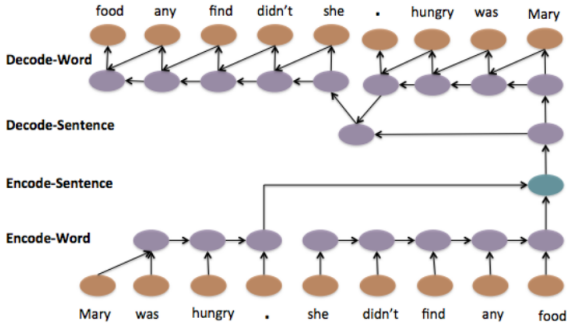

Figure 2: Hierarchical Sequence to Sequence Model.

5 Evaluations

5.1 Dataset

The Signal Media One-Million News Articles Dataset is used for training and testing. It contains 1 million articles that are mainly English, but they also include non-English and multi-lingual articles. Sources of these articles include major ones, such as Reuters, in addition to local news sources and blogs. Each article in the dataset has the following fields.

- **id**: a unique identifier for the article.
- **title**: the title of the article.
- **content**: the textual content of the article (which occasionally contained HTML and JavaScript content).
- **source**: the name of the article source (e.g. Reuters).
- **published**: the publication date of the article.
- **media-type**: either "News" or "Blog".

The dataset contains 265,512 Blog articles and 734,488 News articles. The average length of an article is 405 words.

5.2 Evaluation metric

We use ROUGE-1, ROUGE-2 and BLEU scores on the test set.

ROUGE is a recall-oriented measure widely used in the summarization literature. It measures the n-gram recall between the candidate text and the reference text(s).

$$ROUGE_n = \frac{\sum_{gram_n \in input} count_{match}(gram_n)}{\sum_{gram_n \in input} count(gram_n)} \quad (6)$$

where $count_{match}$ denotes the number of n-grams co-occurring in the input and output.

Purely measuring recall will inappropriately reward long outputs. **BLEU** is designed to address such an issue by emphasizing precision. n-gram precision scores for our situation are given by:

$$precision_n = \frac{\sum_{gram_n \in output} count_{match}(gram_n)}{\sum_{gram_n \in output} count(gram_n)} \quad (7)$$

BLEU first compute the geometric average of the modified n-gram precisions, p_n, using n-grams up to length N and positive weights w_n summing to one. It then combines the average logarithm of precision scores with exceeded length penalization.

BLEU measures how much the words (and/or n-grams) in the machine generated summaries appeared in the human reference summaries. It measures precision.

ROGUE measures how much the words (and/or n-grams) in the human reference summaries appeared in the machine generated summaries. It measures recall.

6 Experiments

6.1 Pre-processing

Minimal pre-processing such as lower-casing, separating on the basis of punctuation marks are applied. All the punctuations are removed in order to reduce the vocabulary size. We also replace tokens which occur less than 10 times with an < unknown > label. Articles with more than 20 % < unknown > labels are safely ignored. For every article, cosine similarity of each sentence present in it, is computed with the corresponding title. We retain only the sentences with top 5 cosine similarity scores per article. To make sure our system is actually summarizing text and not just returning the lead paragraphs, we shuffle the sentences before summarizing. One of the biggest elements of practically implementing our proposed solutions was the preprocessing step. This primarily included pruning out bad examples such as articles with no content or summaries with no content, but also included handling non-unicode characters.

6.2 Constructing Feature Vector

We initially trained a Word2vec model on our whole corpus, which is a two-layer neural net that generates word embeddings. We found out that the results were not good. In order to improve the result, we used a one-hot encoding for our feature vector. This means the size of feature vector will be equal to the vocabulary. The standard LSTM encoder decoder model is implemented in Keras / Theano. Adagrad/Adadelta is used as optimizer for training.

6.3 Training details

The training was performed on a Tesla K-20 GPU with 6GB RAM. The hyperparameters used for training the network is listed in the table below.

Table 1: Hyperparameter list

Hyperparameter	Value
Hidden layer size	500/1000
Batch size	128
Learning rate	0.1
Optimizer	Adagrad/Adadelta

7 Results

7.1 Standard LSTM examples:

Generated:
the new remote control about thicker than ray previous changed ipad apps of case girl s the good thing
Original:
the new remote control is thicker than the previous one and in this case that s a good thing
Generated: (Bad Summary)
to we've difference and diversity we've been by
Original:
where we've succeeded, it's because we've been different.

7.2 Hierarchical LSTM examples:

Generated:
if we spread malicious gossip we our poisoning words own peace demoralise sabotaging situations own influence
Original:
if we spread malicious gossip we are poisoning our own future and sabotaging our own influence with others
Generated:
pressing down on the screen starts different actions depending fathers how much force healing exerted
Original:
pressing down on the screen starts different actions depending on how much force is exerted
Generated:
the misunderstood tried searching i action movies the comedy movies the the remote control
Original:
i also tried searching for action movies and comedy movies with the remote control s microphone
Generated:
bowling ball on third 3rd side with ease redzone left texas went to chris nutall
Original:
bowling ball on third 3rd and in the redzone texas state went to chris nutall senior running back

Table 2: Results for Hierarchical LSTM

No of articles	ROUGE-1 pre	ROUGE-1 rec	ROUGE-1 F1
100	0.1769	0.7561	0.2931
200	0.2372	0.8143	0.3835
500	0.3513	0.7713	0.5104

Table 3: Results for Standard LSTM

No of articles	ROUGE-1 pre	ROUGE-1 rec	ROUGE-1 F1
500	0.1267	0.4113	0.1539

Table 4: Results for Hierarchical LSTM

No of articles	ROUGE-2 pre	ROUGE-2 rec	ROUGE-2 F1
100	0.1131	0.4997	0.1969
200	0.1428	0.6014	0.2309
500	0.2828	0.5945	0.4222

7.3 Hierarchical LSTM summary 1

Original Text:(It's preprocessed text)
here are some first impressions of the iphone 3d touch capability the iphone signature new feature pressing down on

the screen starts different actions depending on how much force is exerted when you press down hard the device responds with a tapping sensation i found the new iphones pretty straightforward to use the ipad pro is capable of running two full size ipad apps on the screen at the same time the big screen ipad also felt comfortable to hold it weighs about grams which is the same weight as the very first ipad the apple tv is significantly different from past models the software interface has a white background and vivid colours similar to apple tv music app the remote also includes motion sensors for gaming the new remote control is thicker than the previous one and in this case that is a good thing

Summary:
here are some impressions of iphone products capability pressing down on the screen starts different actions depending on how much force is exerted the big screen ipad capable felt comfortable two hold same weight as the very first ipad the tv software which is significantly same vivid colours models the new remote control is thicker than the previous

7.4 Hierarchical LSTM summary 2

Original Text:(It's preprocessed text)
an email chain between former secretary of state hillary clinton and then commander of united states is raising questions the chain included clinton and petraeus getting acquainted as well as some personnel matters according to the officials the chain begins on january and ends on feb and contains less than 10 emails in total between clinton and petraeus the official said these emails are now in our possession and will be subject to freedom of information act requests furthermore we asked the ig to incorporate this matter into the review secretary kerry requested in march we have also informed congress of this matter the official said the emails were found because they have now been digitized and are easier to search clinton turned over to state her work related emails from the server and kept emails she deemed personal the fbi is looking into whether any classified materials were mishandled

Summary:
email chain between former secretary of state hillary clinton and the commander of united states the chain begins on january and ends and feb the contains less than emails between clinton and petraeus the official said furthermore we asked the ig to incorporate related matter into the review the fbi over looking into whether any classified materials were mishandled

Table 5: Results for Standard LSTM

No of articles	ROUGE-2 pre	ROUGE-2 rec	ROUGE-2 F1
500	0.0892	0.2591	0.1004

Table 6: Results for Hierarchical LSTM

No of articles	ROUGE-3 pre	ROUGE-3 rec	ROUGE-3 F1	BLEU
100	0.039	0.2043	0.0079	0.1145
200	0.065	0.2804	0.1075	0.2089
500	0.1565	0.4465	0.3876	0.3702

Table 7: Results for Standard LSTM

No of articles	ROUGE-3 pre	ROUGE-3 rec	ROUGE-3 F1	BLEU
500	0.002	0.1647	0.0059	0.007

8 Conclusions and Work in Progress

In terms of NLP, most problems can be transformed to the problem of getting representations of language elements (words, sentences, paragraphs, and documents). We have completed the implementation of Standard LSTM. Our next step is to implement the hierarchical LSTM encoder-decoder model for abstractive document summarization.

9 References

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

[2] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[3] Thang Luong, Ilya Sutskever, Quoc V. Le, Oriol Vinyals, and Wojciech Zaremba. Addressing the rare word problem in neural machine translation. *CoRR*, abs/1410.8206, 2014.

[4] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112, 2014.

[5] Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey E. Hinton. Grammar as a foreign language. *CoRR*, abs/1412.7449, 2014.