Sentence Fusion
Supervised Project

Andrés HERRERA CEPERO Laurine JEANNOT
Quentin POUVREAU Imane ZOUHRI

June 1, 2018

Université de Lorraine

Supervised by Claire Gardent
Year 2017-2018

Host Organization: Loria

Project “Sentence Fusion”

Authors: Andrés Herrera Cepero Laurine Jeannot Quentin Pouvreau
Imane Zouhri

Supervised by Claire Gardent

Academic year: 2017-2018

Contents

1 Introduction

Sentence fusion is a text-to-text generation task which takes related sentences as input and merges them into a single output sentence. It is typically used in the context of multi-document summarization. It for example allows the generation of a summary from multiple news articles that contain overlapping information. It is also used for the production of abstracts where simple sentences need to be fused into a single more readable phrase. Another application allows to make more sophisticated Question/Answer systems, where multiple responses to a single question can be provided as a single phrase.

The aim of this supervised project is to investigate deep learning approaches to sentence fusion using the Split-and-Rephrase dataset [Gardent and al, 2017][1], and to evaluate them.

2 Notions

From the perceptron to sequence to sequence models, some background reading around the deep learning landscape was required in order to have a better understanding of the task at hand.

2.1 Perceptron & Sigmoid neurons

Perceptron: Artificial neurons are inspired from biological neurons, they take several binary values as an input and then return a binary value in output. Inputs are weighted according to their importance, so the perceptron weight up inputs and compare the result to a threshold value : if the result is greater than the threshold value the perceptron's output is 1, else it is 0. [12]

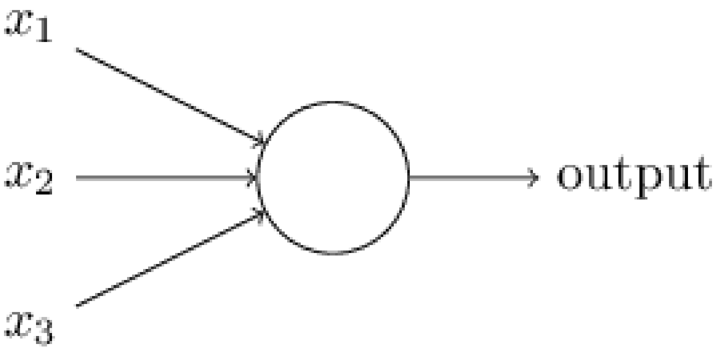


Figure 1: A perceptron

Sigmoid neurons : They are an improvement on the perceptron since they output values between 0 & 1, which allows to apply it to more applications.

2.2 Neural Networks

Neural Networks is a biologically-inspired programming paradigm which enables a computer to learn from observational data.[12] This approach combines sigmoid neurons to create a network.

The system as by its name is inspired from the actual brain neural network and is getting optimized by several learning methods. When talking about neural networks it is generally feed forward networks that most people would think of. Nevertheless, they can also be composed of recursive neurons, explained below. Neural networks can be used in many different ways, for tasks such as sentence fusion but also for image recognition, image classification and Artificial Intelligence in games.

2.3 Deep Neural Networks

Deep learning is a powerful set of techniques for learning in neural networks. Deep Neural Networks (Deep NN) is a neural network with many hidden layers. This fact marks the only difference with "non deep" neural networks as illustrated below :

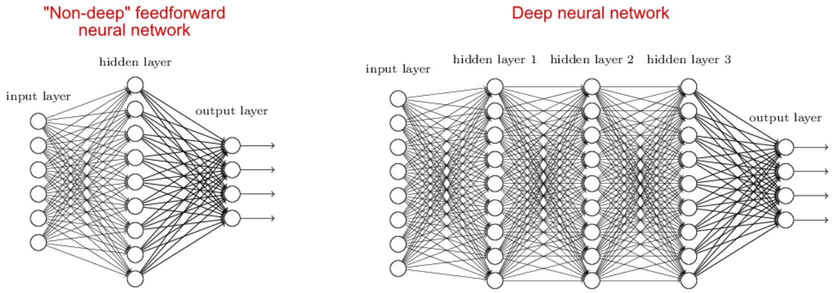


Figure 2: Difference between NN and Deep NN [2]

When training neural networks, an epoch represents the number of times a forward pass and a backward pass is applied to all the training data and the hyperparameters are updated. We can also represent an epoch as follows [3] :

$$\boxed{\text{1 epoch } = \text{1 forward pass} + \text{1 Backward pass for all the training samples}}$$

2.4 Recurrent Neural Networks and LSTM

Recurrent Neural Networks (RNN) are neural networks that are specialized for processing a sequence of values x(1)...x(T) [4]. At each timestep neurons from the hidden layer take in both the input X*t* (a vector representation of a word) and the state from the previous timestep h*t-1*. In this model a prediction (Yt) will be made at each timestep [5].

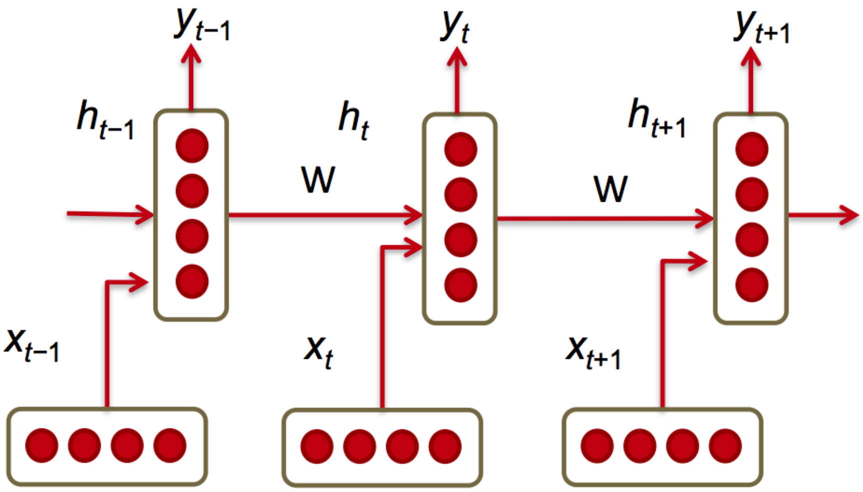


Figure 3: RNN Diagram [15]

Long-short-term memory or more commonly known LSTMs are a building unit for layers of recurrent neural network. They are used to classify, process and remember the information over time and trough time steps plus it also has the ability forget certain information. The dimensions of the inputs and outputs need to be known, so the architecture of the long-short-term-memory(LSTM) can solve a seq2seq problem[6].

2.5 Sequence to Sequence

A sequence to sequence model (seq2seq model), also known as an encoder-decoder model, is a many-to-many RNN architecture that consists of two recurrent RNN. This model is different to the RNN model from the previous section in that predictions will be made only by the decoder and not at each timestep, which makes it a good fit for machine translation. In fact, this architecture is used by the Google's translate service. The interesting part of the encoder is the final hidden state, which contains the information from all the input sequences and outputs a single vector. On the other hand, the decoder takes the vector from the encoder, considers it as the starting state and produces an output sequence as portrayed in the figure below.

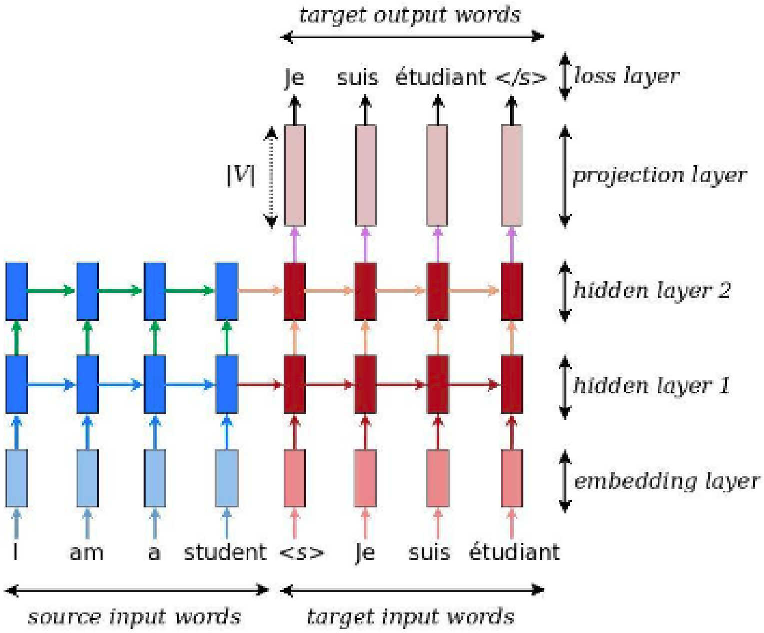


Figure 4: Seq2Seq

2.6 Sequence to Sequence Models and Deep Learning

During the investigation we wondered whether the usage of the term "deep learning" is accurate in the context of sequence to sequence models. Conversations with our supervisor revealed that when using sequence to sequence models there is a tendency towards shallower networks rather than deep, despite the fact that in the past a lot of layers were normally used. This fact is reinforced by the statement from the director of Facebook AI research and influential figure on deep learning, Yann LeCun, when he proclaimed that "deep learning is dead" [**?**].

3 Our Task

As mentioned in the introduction, the aim of this project is the investigation and evaluation of deep learning approaches to sentence fusion using the split-and-rephrase dataset. The deep learning model in which we have focused on is a sequence to sequence model that takes as input a sequence of related sentences and provides a prediction of its related complex sentence as test output.

3.1 Sequence to Sequence Approach

Because of the sequence based nature of the sentence fusion task, a sequence to sequence model was the best suited to utilize in the investigation. Since its beginnings, different approaches have been used to sequence to sequence tasks, some focusing on statistics. Recently, however, Neural Machine Translation(NMT) apply a sequence to sequence translation which produces higher quality results. For this we used OpenNMT, a general-purpose attention-based seq2seq system from MIT used for neural sequence modeling and neural machine translation " [6].

Why OpenNMT ? OpenNMT was chosen because it gives access to a ready-made sequence to sequence system that we could use to produce sentence fusion models.

The OpenNMT release chose, OpenNMT-py, is implemented on top of Pytorch. We choose OpenNMT-py because we had a course this semester on deep learning in which we learn to use PyTorch. Moreover python language was the most friendly for our team.

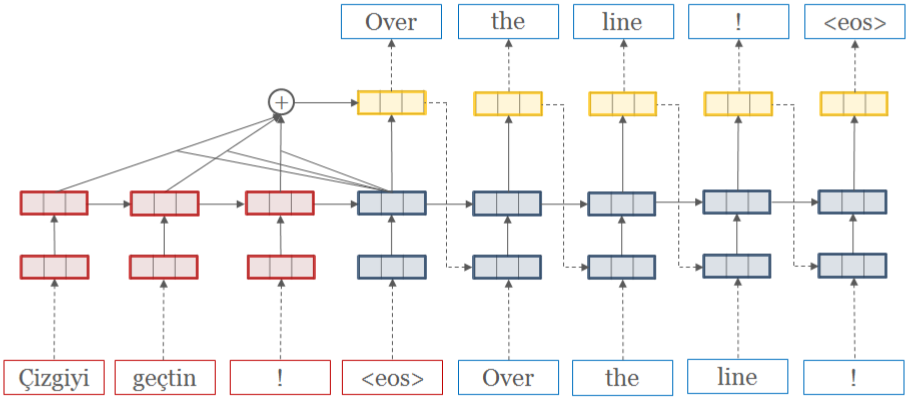


Figure 5: OpenNMT

The parameters for the default OpenNMT pytorch model which we use for producing our models are :

- The size of the input : Which is 500 and represent the number of neurons of the input,
- The hidden size : Which is also 500 and represent the number of neurons of a decoder layer,
- The Number of layers is 2,
- The initialization of the dropout probability is equal to 0.3. This number allows a sequence to have the same dropout mask for different time steps for consistent masking.

3.2 Data

We used the split-and-rephrase datasets (benchmark-v0.1 and benchmark-v1.0 [14]). A dataset is composed of 6 files : three files for complex sentences and three other files for simple sentences, a training data file, a testing data file and a validation data file. The *train.complex, test.complex, validation.complex* data files are composed of complex sentences that are repeated on many lines to make pairs with the simple sentences. The *train.simple* data file is composed of lines that contain a small number of simple sentences, as the *test.simple* and the *validation.simple* are. The simple data files have the same number of lines than the number of sentences from the complex data files. By combination, a complex file and its corresponding simple file form a set of complex-simple pairs.

Number of pairs

Version	Train lines	Test lines	Validation lines
benchmark-v0.1	886857	81309	97951
benchmark-v1.0	1331515	43958	40879

Each complex sentence can be produced by many configurations of simple sentences, that is why there are few distinct complex sentences compared to the number of complex-simple pairs.

Distribution

Version	Distinct complex sentences	Complex-simple pairs
benchmark-v0.1	5546	1098221
benchmark-v1.0	18830	1445159

As an example, both of the following configurations of simple sentences produced this complex sentence :

The American , Duncan Rouleau created the character of Baymax who appeared in the film Big Hero 6 which starred Alan Tudyk .

Baymax was created by Duncan Rouleau .
Duncan Rouleau is an American .
Baymax is a character in Big Hero 6 .
The film , Big Hero 6 , starred Alan Tudyk .

Duncan Rouleau is American .
Created by Duncan Rouleau , Baymax , is a character in Big Hero 6 .
The film , Big Hero 6 , starred Alan Tudyk .

Sentence Fusion can make some words disappear from the vocabulary. That is why the vocabulary size from simple data is bigger than the vocabulary size from complex data.

I.e for benchmark-v0.1 :

Data	Simple data	Complex data
vocabulary size	2963	2739

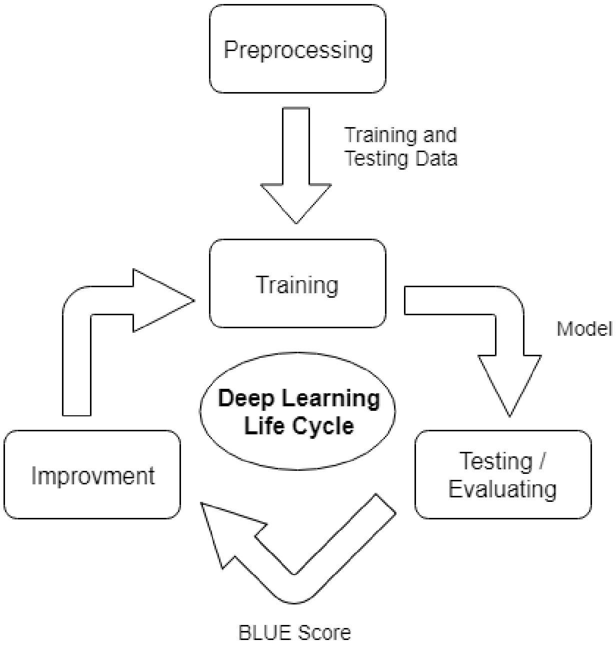


Figure 6: Process of the project

3.3 Work

The chosen way to investigate deep learning approaches to sentence fusion has been through the production and evaluation of sequence to sequence sentence fusion models. In order to produce such models, powerful hardware was required that could process a great amount of data in an acceptable time. After struggling to set up a personal machine for deep learning, the solution found was using Grid 5000, a distributed computing solution which allowed using multiple powerful GPU enabled nodes for our task.

Using OpenNMT-py: As we got enough preprocessed data, as explained formerly, we can train a model. Once we have a model we can test it. During the test phase we feed the model with simple sentences and it makes the fusion and returns predictions, i.e.(the corresponding complex sentences). The accuracy of the model's predictions is then evaluated with the BLEU score and the model will be improved until we consider it efficient enough.

3.3.1 Environment Setup

The setup process followed is the following:

1. Sign up to Grid 5000 with the University e-mail,
2. Create SSH key in a local computer and upload it to Grid 5000,
3. Install Deep Learning framework and requirements in the node assigned,

4. Upload data files and any shell script,
5. Setup is ready.

3.3.2 Grid5000 usage

Since Grid5000 is the hardware we used for carrying out our deep learning investigations, there are a number of commands which we found vital to successfully use it:

1. Connecting to grid5000 and navigating to OpenNMT-py:

```
ssh gridusername@access.grid5000.fr

ssh  nancy

grid500username@fnancy:~$  cd  seq2seq/OpenNMT-py
```

2. Uploading files from and downloading files to a personal computer

```
scp  localfiletoupload gridusername@access.grid5000.fr:nancy/seq2seq/OpenNMT-py

scp  gridusername@access.grid5000.fr:nancy/seq2seq/openNMT-py/serverfiletodownload.extension .
```

3. Running Jobs

The initial node to which one is redirected after running the ssh nancy command doesn’t generally have a GPU. Therefore the following command allows running a job on a GPU enabled node.

```
oarsub -I -p "GPU!=’NO’" -q production
```

This command provides exclusive use of the node to the person that requests it so it must be used sensibly. For example for preprocessing or training around 10 000 lines or running those tasks that will take around an hour.

In order to run longer tasks it is required to "make a node reservation". This enables to reserve a single node or combine the power of a number of Grid5000 nodes to make the task run faster. Reservations are automated which means that they can be done through a command. Specific care must be given to assign enough walltime to run the task since it is not possible to extend it after submition.

The preprocess, training and testing commands which allow obtaining a model and getting predictions are included in a shell script.

```
oarsub -q production -p "cluster=’grele’" -l "nodes=3,
walltime=24:00:00" ./filename_of_shell_script.sh
```

4. Checking Jobs status and deleting a job

Once a reservation has been submitted an id is assigned to the job and a waiting status is given. Using the following command allows to know the status as well as an estimated date and time for when the task will start running.

```
oarstat -fj <JOBID>
```

To delete a job use

```
oardel <JOBID>
```

The commands in this section allowed making use of Grid5000 for our task as depicted in the figure below:

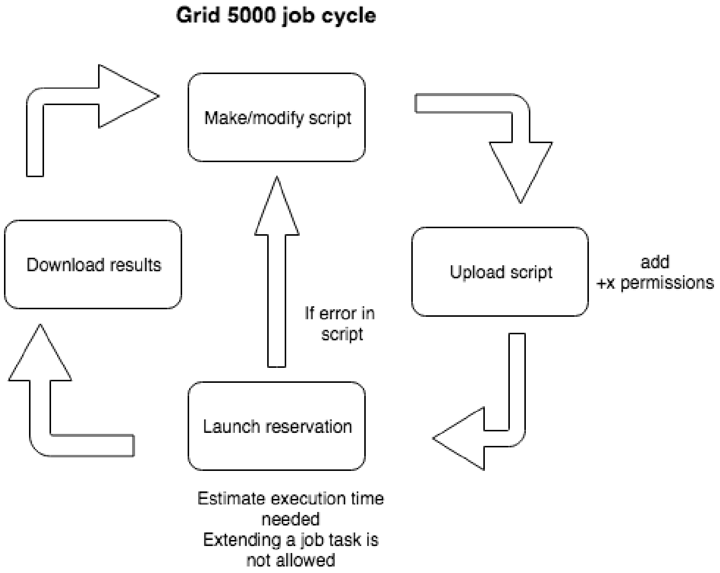


Figure 7: Grid 5000

3.3.3 Preprocessing

Machine learning models are produced by training them with large amounts of data. Preprocessing covers any manipulation this data goes through that prepares it for training. Preprocessing has been done at different levels in the project :

The six files mentioned in the data section are produced by a script present in the split and rephrase folder. The first baseline model was to be trained with 10 000 simple - complex pairs. Therefore training and validation files were reduced to the first 10 000.

Another part where preprocessing was needed was at the time of obtaining predictions from training data. Since the training data files are 878 000 lines long predictions would take too long if used raw, therefore train.simple was cut to 81 308 lines.

Translating the S&R dataset to other languages and training a model with the new data was attempted.

The first approach was directly translating "final-complexsimple-meanpreserve-intreeorder-full.txt", a 20 000 000 line file used by the Split and Rephrase script that produces the training data.

Googletranslate API was used for this purpose. Even if the API mentioned there were no limits in translation it stopped every a few thousand lines, despite the blocking 30,000 lines were translated. This translation could not nevertheless be used since the file is paired with another json file that manages the training/testing/validation splits and we couldn't find how to modify this last one. Therefore the whole file needed to be translated.

Given the difficulty of translating the file an alternative approach taken was to translate the WebNLG benchmark. This benchmark was used to produce the Split and Rephrase dataset so it contains its sentences and share identification ids. Therefore the approach was to translate the benchmark sentences and replaced them by ids on the target translation file.

The WebNLG benchmark was successfully translated to Spanish. A prototype script to replace just the complex sentences on the target script with the translated ones was produced. The script presented two stages, one where all the file is saved into a data structure and another stage where the file is transversed and the relevant sentences are replaced with the translations.

The estimated running time for the first stage is 7 days.

3.3.4 Training & Testing

As mentioned in the preprocessing section an initial model was obtained from 10,000 training lines. The bleu score obtained, 3, was very low and it was concluded that the reason for that was that not enough training data was used. A new model was then trained with the full dataset, 870,000 lines for 13 epochs and a similar score was obtained. It was then retrained for 20 epochs and the score raised to 5,6. In order to prove the relationship between number of epochs and the obtained bleu score, another model was trained for 60 epochs. This last model gave better scores on 13 epochs. Still aiming to improve the score we trained a new model on the full data using the attn_copy mechanism.

In order to do the training and predictions OpenNMT commands needed to be written in a bash script for its execution on Grid5000.

The training

4 Results

In this section we will talk about the evaluation of our work, *i.e* the BLEU score and the manual evaluation.

4.1 BLEU score

"The closer a machine translation is to a professional human translation, the better it is" [10].
This is the global principle of what is the BLEU score. In other words, the BLEU score is a comparison between a human translation and the one made by a machine. To do that it will compare the number of n-grams (usually from 1 to 4) that a given sentence share with the aligned sentence in the reference corpus. The score is computed for each sentence and at the end we will average all these results to obtain a general score. Higher the score is, better is the translation.

The task her is not to evaluate a translation made by a machine but the fusion of several sentences. The reference corpus here is a data set of complex sentences extract from Split-and-Rephrase . Thus, it is also relevant to compare the complex sentences we obtained in output of the script to the the corpus of complex sentences.

After the first training of our model on part of the training data, 10,000, we obtained predictions by feeding the model a small part of the *test.simple* file. At this moment the score was quite low. Then we did another training with more data to feed the model and we compared our predictions on this model with the corresponding part of the *test.complex* file. At this moment we obtained a BLEU score of 11.63.

Then we ran the training process on the whole test data (about 81,000 sentences) using different numbers of epochs, we obtained predictions for the entire *test.simple* file. In order to obtain the best result for the blue score, and since we have noticed that it changes according to the computer. The BLEU score from one node was different even if the same model was used, we have decided to evaluate it many times in order to get the best one. The results of these trainings are presented in the graph in figure 7. These BLEU scores was calculate with the training on the dataset 0.1 from Split-and-Rephrase. In the figure 8 there are the evaluations for the dataset 1.0.

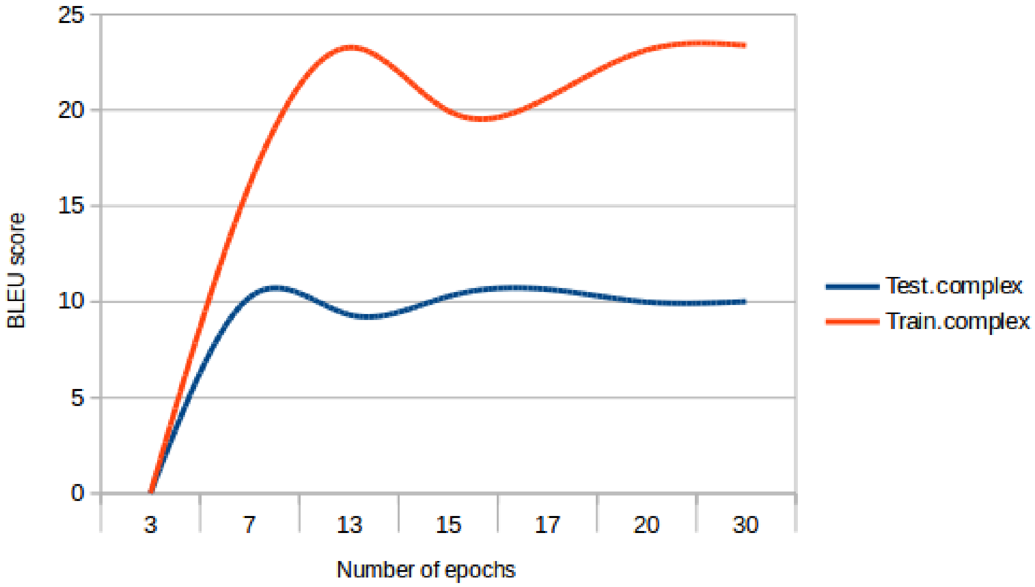


Figure 8: BLEU score for fullmodel S&R v.0.1

To illustrate the results we obtained with the part of the training dataset, here are examples of what we obtained:

Epochs	Dataset	Reference sentence	Produced sentence	Score
7	Test	The 11th Mississippi Infantry Monument -LRB- established in 2000 -RRB- is located in Gettysburg , Pennsylvania , and falls under the category of Contributing property .	The 11th Mississippi Infantry Monument , is located in 2000 , Pennsylvania , Pennsylvania , Pennsylvania , Adams County , Adams the United States .	24.39
13	Train	The 11th Mississippi Infantry Monument , established in 2000 , is a contributing property and located in Gettysburg and Adams County in Pennsylvania .	The 11th Mississippi Infantry Monument , is located in 2000 , Adams County , Pennsylvania .	31.03

As neural networks parameters are randomly initialized, the results may be very different from a training to another, even when training on the same data. Here these results are the best ones we obtained for the last training, and it is a coincidence that

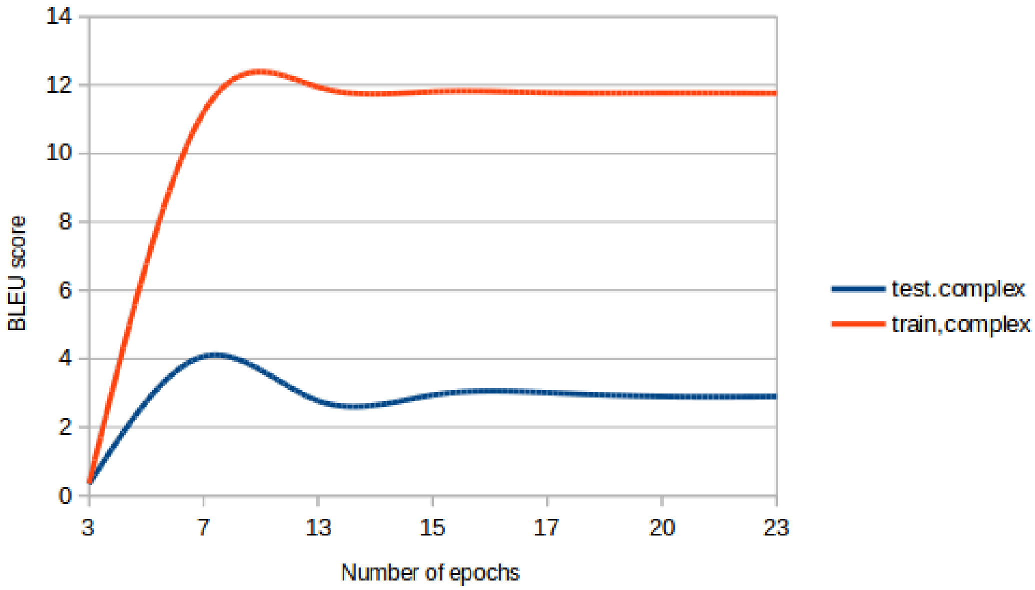


Figure 9: BLEU scores for fullmodel_copy_attn v.0.1

the best of the test dataset is about the same subject as the train dataset.

4.2 Manual evaluation

Give a global evaluation of the task is good, but evaluate it in details is better. The aim of the manual evaluation is to give an idea of how good is the predictions for each sentence. Hence we had to establish rating criteria to have an homogeneous evaluation.

We have defined four main criteria for the evaluation:

- If the sentence is grammatically correct, graded from 0 to 2
- The fluency of the sentence graded from 0 to 5
- If the sentence produced is a complex sentence or not
- If we can guess the theme of the sentence or not

Example of sentence produced by our first training :

The Baku Turkish Martyrs Memorial , located in Azerbaijan , is dedicated to the Ottoman Army soldiers killed in the Battle of Baku .
Evaluation: Grammatical: 2; Fluency: 5; Complex: 1

Other examples from 13 epochs with training dataset:

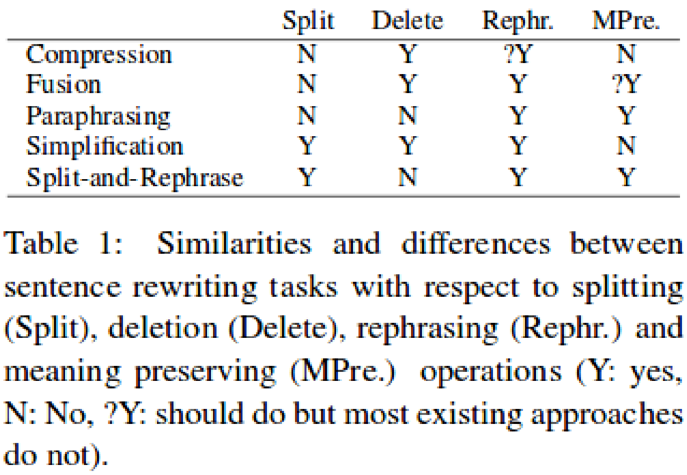
The best sentence the training produced: *The 11th Mississippi Infantry Monument is located in 2000 , is located in 2000 , Adams County , Pennsylvania* .(Score of 31.03)
Evaluation: Grammatical: 2; Fluency: 5; Complex: 1

The worst sentence: *A , was a* (Score of 0.01)
Evaluation: Grammatical: 0; Fluency: 0; Complex: 0
In fact this case is difficult because it is not a sentence at all. So should we evaluate it as a sentence? Here we decided to evaluate it as it.

5 Discussion

The aim of this project was to investigate and evaluate deep learning approaches to make sentence fusion using Split-and-Rephrase as the dataset.

Our expectations were that a model trained on the full data would produce at least complex sentences preserving the meaning/topic from the simple ones. It however mostly produced phrases which although had some words in common with the simple ones, the meaning was not preserved.
Nevertheless, according to table 1 (figure 10) in the Split-and-Rephrase paper, most current approaches to sentence fusion have this problem.



	Split	Delete	Rephr.	MPre.
Compression	N	Y	?Y	N
Fusion	N	Y	Y	?Y
Paraphrasing	N	N	Y	Y
Simplification	Y	Y	Y	N
Split-and-Rephrase	Y	N	Y	Y

Table 1: Similarities and differences between sentence rewriting tasks with respect to splitting (Split), deletion (Delete), rephrasing (Rephr.) and meaning preserving (MPre.) operations (Y: yes, N: No, ?Y: should do but most existing approaches do not).

Figure 10: Table 1 Split and rephrase research paper [1]

The scores were not as high as anticipated. The bleu score of the main model was not very high and the prediction results of the model with the copy attention were supposed to be better than without it, but in our case it was the opposite. Grid5000 permitted us to reduce the execution time for the training and predicting phases of our models, however the limited number of nodes or its high demand meant there was a lot of waiting time before a job could be executed. Ideally, the manual evaluation would be done on about 20,000 sentences, and evaluated once by two different persons. Therefore, we achieved to do the evaluation on 1,000 sentences from the predictions from the training dataset without the copy option with 13 epochs.
The bleu scores we obtained are very low compared to the ones that are in the Split

and Rephrase: Better Evaluation and a Stronger Baseline paper[13]. Our bleu scores are around 10, while the least on the paper is 39.97. During testing, as expected, the predictions from simple sentences present in the training set had a higher bleu score than those predictions from simple sentences of the test set, unseen by the model.

6 Conclusion

Following the results of the investigation there are two possible conclusions : On the one hand we could conclude that deep learning is a good approach to sentence fusion but very specialized knowledge is needed in order to obtain good results. This is supported by the good scores on the split and rephrase task with the same data.

On the other hand, we could conclude that even deep learning had good results on the Split-and-Rephrase project, it is not as good a fit for sentence fusion since the attention model didn't provide better scores whereas it did it on the Split and Rephrase task. In which case maybe sentence fusion needs more data or a different approach to make good predictions.

By all means we can conclude that deep learning for sentence fusion is a hard and time consuming task, but further investigation to explore each of the two options just mentioned is recommended.

References

[1] https://hal.inria.fr/hal-01623746/document.

[2] https://stats.stackexchange.com/questions/182734/what-is-the-difference-between-a-neural-network-and-a-deep-neural-network-and-w/.

[3] https://www.mathworks.com/matlabcentral/answers/62668-what-is-epoch-in-neural-network.

[4] http://www.deeplearningbook.org/contents/rnn.html.

[5] http://cs224d.stanford.edu/lecture_notes/notes4.pdf.

[6] http://opennmt.net/.

[7] https://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf.

[8] Benoît,Favre, *Deep Learning for Natural Language Processing*, http://pageperso.lif.univ-mrs.fr/ benoit.favre/dl4nlp/slides/02-deep-learning.pdf.

[9] Josh,Meyer, *A Tensorflow Tutorial*, https://www.tensorflow.org/tutorials/seq2seq.

[10] Papineni Kishore, Roukos Salim, Ward Todd, and Zhu Wei-Jing, *BLEU, a Method for Automatic Evaluation of Machine Translation*, http://aclweb.org/anthology/P/P02/P02-1040.pdf.

[11] Yann Lecun, *"OK, Deep Learning has outlived its usefulness as a buzz-phrase. Deep Learning est mort. Vive Differentiable Programming![...]Other concepts will be needed for that, such as what I used to call predictive learning and now decided to call Imputative Learning. More on this later.."*, 2018.

[12] Michael,Nielsen, *Neural Networks and Deep Learning*, http://neuralnetworksanddeeplearning.com/chap1.html.

[13] yoav goldberg roee aharoni, *Split and rephrase: Better evaluation and a stronger baseline*, https://arxiv.org/pdf/1805.01035.pdf.

[14] Narayan Shashi, Gardent Claire, B. Cohen Shay, and Shimorina Anastasia, *Split and rephrase*, https://github.com/shashiongithub/Split-and-Rephrase.

[15] Richard Socher.

[16] Viswanathan, P and Krishna, P Venkata, *Text fusion watermarking in medical image with semi-reversible for secure transfer and authentication*, Advances in Recent Technologies in Communication and Computing, 2009. ARTCom'09. International Conference on, IEEE, 2009, pp. 585–589.

A Appendix

A.1 Statistics about the results

fullmodel-v0.1: model trained on full Split-and-Rephrase dataset v.0.1

fullmodel-v0.1-copy-attn: model trained on full Split-and-Rephrase with copy attention mechanism

Remark: Min and Max are respectively the minimum and the maximum scores got among the predicted sentences, and Median is the median got among the set of scores for each sentence.

Dataset	Model name	Reference data	N. epochs	Min	Max	BLEU score	median
v.0.1	fullmodel-v0.1	Test.complex	3	0,000143	19,70	0.10	0,24
			7	0,06	36,59	8.97	3,64
			13	0,14	50,45	17.60	2,93
			15	0,14	50,45	12.85	2,56
			17	0,14	50,45	11.56	2,44
			20	0,14	50,45	13.08	2,53
			23	0,14	50,45	13.08	2,53

Dataset	Model name	Reference data	N. epochs	Min	Max	BLEU score	median
v.0.1	fullmodel-v0.1	Train.complex	3	0,000278	34,90	0.02	0,01
			7	0,03	46,60	16.14	22,60
			13	0,01	44,43	23.27	18,71
			15	0,01	37,43	19.96	18,54
			17	0,01	46,10	20.65	18,02
			20	0,01	35,96	23.14	20,51
			23	0,01	35,96	23.38	20,51

Dataset	Model name	Reference data	N. epochs	Min	Max	BLEU score	median
v.0.1	fullmodel-v0.1-copy-attn	Test.complex	3	0,07	14,47	0.05	1,64
			7	0,06	36,43	5.83	1,76
			13	0,02	30,73	3.16	1,56
			15	0,02	29,33	3.65	1,56
			17	0,02	28,95	3.83	1,59
			20	0,02	28,95	3.57	1,56
			23	0,02	28,95	3.59	1,56

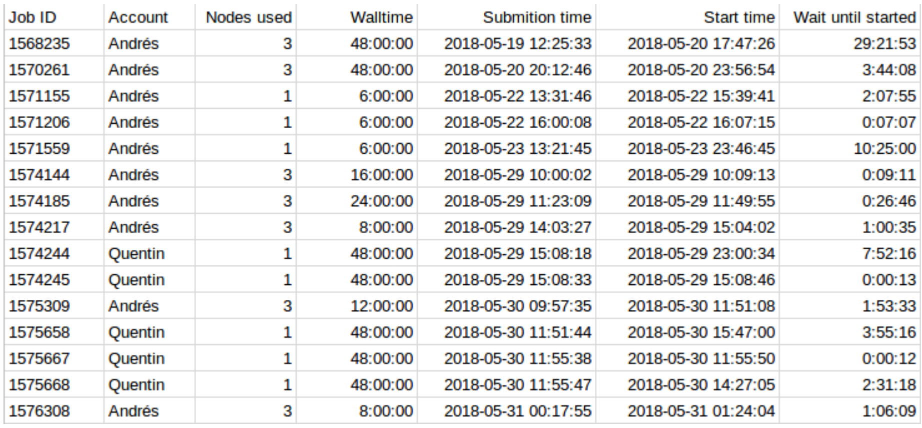
Dataset	Model name	Reference data	N. epochs	Min	Max	BLEU score	median
v.0.1	fullmodel-v0.1-copy-attn	Train.complex	3	0,05	30,27	0,36	1,03
			7	0,02	65,11	11,21	4,16
			13	0,02	59,54	11,94	6,07
			15	0,02	59,54	11,81	6,07
			17	0,02	59,54	11,78	6,07
			20	0,02	59,54	11,77	6,07
			23	0,02	59,54	11,76	6,07

Dataset	Model name	Reference data	N. epochs	Min	Max	BLEU score	median
v.1.0	fullmodel-v1.0	Test.complex	3	0,18	5,15	0.03	0,50
			7	0,02	28,62	0.31	1,06
			13	0,05	27,67	0.31	1,06
			15	0,05	33,40	0.36	1,06
			17	0,05	33,40	0.35	1,06
			20	0,05	33,40	0.35	1,06
			23	0,05	33,40	0.35	1,0

A.2 Repository address

https://gitlab.com/ClaireGardent/PT2018_Sentence_Fusion.git

A.3 Waiting time for jobs reservation on Grid5000



Job ID	Account	Nodes used	Walltime	Submition time	Start time	Wait until started
1568235	Andrés	3	48:00:00	2018-05-19 12:25:33	2018-05-20 17:47:26	29:21:53
1570261	Andrés	3	48:00:00	2018-05-20 20:12:46	2018-05-20 23:56:54	3:44:08
1571155	Andrés	1	6:00:00	2018-05-22 13:31:46	2018-05-22 15:39:41	2:07:55
1571206	Andrés	1	6:00:00	2018-05-22 16:00:08	2018-05-22 16:07:15	0:07:07
1571559	Andrés	1	6:00:00	2018-05-23 13:21:45	2018-05-23 23:46:45	10:25:00
1574144	Andrés	3	16:00:00	2018-05-29 10:00:02	2018-05-29 10:09:13	0:09:11
1574185	Andrés	3	24:00:00	2018-05-29 11:23:09	2018-05-29 11:49:55	0:26:46
1574217	Andrés	3	8:00:00	2018-05-29 14:03:27	2018-05-29 15:04:02	1:00:35
1574244	Quentin	1	48:00:00	2018-05-29 15:08:18	2018-05-29 23:00:34	7:52:16
1574245	Quentin	1	48:00:00	2018-05-29 15:08:33	2018-05-29 15:08:46	0:00:13
1575309	Andrés	3	12:00:00	2018-05-30 09:57:35	2018-05-30 11:51:08	1:53:33
1575658	Quentin	1	48:00:00	2018-05-30 11:51:44	2018-05-30 15:47:00	3:55:16
1575667	Quentin	1	48:00:00	2018-05-30 11:55:38	2018-05-30 11:55:50	0:00:12
1575668	Quentin	1	48:00:00	2018-05-30 11:55:47	2018-05-30 14:27:05	2:31:18
1576308	Andrés	3	8:00:00	2018-05-31 00:17:55	2018-05-31 01:24:04	1:06:09