Neural Attentive Weighting for Concept-to-Text Natural Language Generation

Pradipta A.B. Hendri, Matt Proetsch, and Daniel Volk

School of Information, University of California, Berkeley
{dip,mproetsch,daniel.volk}@berkeley.edu

April 23, 2018

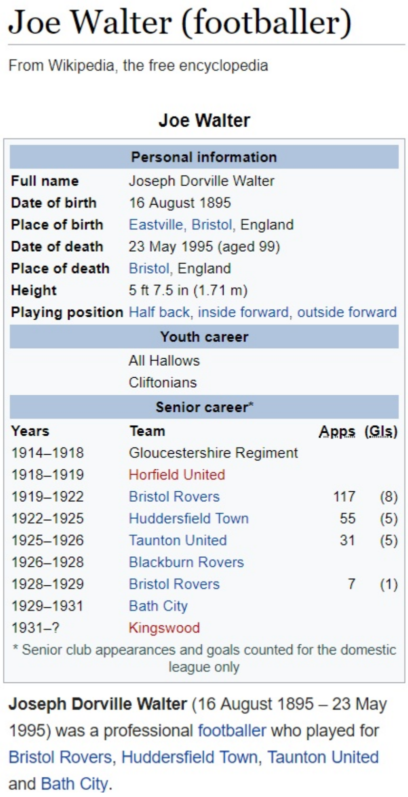
Abstract

This paper proposes enhancements to design of convolutional sequence-to-sequence neural frameworks to solve the task of concept-to-text generation for the Wikipedia biography dataset. The network will take contextual data from the infoboxes provided on Wikipedia's biography pages and generate the first sentence of the biography. The paper incorporates context embeddings along with an encoder-decoder framework for sequence generation. The model structure is similar to recent works in machine translation which take advantage of convolutional encoder-decoder networks with attentive weighting in order to generate our text. These models have shown promising performance in comparison to recurrent neural networks, while requiring less time to train due to parallelism compatible with GPUs. The models are assessed against baseline models as well as previous works via BLEU and perplexity scores.

1 Introduction

CONCEPT-TO-TEXT generation takes structured data such as weather data or stock tickers and attempts to translate it into natural language. Development of these methods can be very useful for automating the generation of weather reports and stock market news and presenting data in a natural way, and appropriate for subsequent processing with text-to-speech systems.

The task we are focusing on for our model is creating biographies from Wikipedia infoboxes. Infoboxes are fact tables included on the subject's wikipedia biography page, often containing information such as name, birthday, occupation, awards/honors, etc. From this data our model will generate the first sentence of the Wikipedia article biography such as in Figure 1 below.



Joe Walter (footballer)

From Wikipedia, the free encyclopedia

Joe Walter

Personal information	
Full name	Joseph Dorville Walter
Date of birth	16 August 1895
Place of birth	Eastville, Bristol, England
Date of death	23 May 1995 (aged 99)
Place of death	Bristol, England
Height	5 ft 7.5 in (1.71 m)
Playing position	Half back, inside forward, outside forward
Youth career	
	All Hallows
	Cliftonians

Senior career*			
Years	**Team**	**Apps**	**(Gls)**
1914–1918	Gloucestershire Regiment		
1918–1919	Horfield United		
1919–1922	Bristol Rovers	117	(8)
1922–1925	Huddersfield Town	55	(5)
1925–1926	Taunton United	31	(5)
1926–1928	Blackburn Rovers		
1928–1929	Bristol Rovers	7	(1)
1929–1931	Bath City		
1931–?	Kingswood		

* Senior club appearances and goals counted for the domestic league only

Joseph Dorville Walter (16 August 1895 – 23 May 1995) was a professional footballer who played for Bristol Rovers, Huddersfield Town, Taunton United and Bath City.

Figure 1: Wikipedia infobox for Joe Walter, and the corresponding first paragraph

Many of the biography sentences follow a similar structure to the one above. That structure being `name_1 name_2 ( birthdate_1 birthdate_2 birthdate_3 - deathdate_1 deathdate_2 deathdate_3 )`. While this is a useful visualization, the contextual structure we employ takes into account the word, infobox field, start and end positions as well as global embeddings. We will discuss this later on, but the first word in the infobox would be encoded as `(Frederick, (name, 1, 3))`, indicating the word, field name, as well as the start and ending positions.

To address this problem we will use a context embedding process [1], combined with sequence-to-sequence architectures [2], and self-attentive weighting [3]. The context embedding will take both local and global context into consideration.

Traditional concept-to-text tasks include WEATHERGOV and ROBOCUP both of which have tens of thousands of observations and less than 1K word vocabularies. The Wikipedia biography dataset was gathered by [1] and contains over 700K samples and 400K unique words. The vocabulary is extremely diverse due to the fact that there are many names and tokens that could be very specific to particular individuals.

Table 1: Data Description

	Avg.	Percentile		
		5%	50%	95%
words/sent	26.1	13	24	46
infobox words/sent	11.6	4	11	23
words/infobox	45.8	13	39	101
fields/infobox	12.4	5	12	21

2 Background

In recent years, neural language models have outperformed traditional rule-based approaches when it comes to the task of text generation. Recent advances in computing power have allowed for more complex modeling to be used such as LSTMs and convolutional neural networks.

While recurrent structures have become widely used for machine translation tasks, recent work in the field of text generation has focused on sequence-to-sequence convolutional frameworks with attention mechanisms. These have proven successful and are often faster to train than a recurrent network. Gehring et al. (2017) [2] outperformed the previous best results in English-Romanian translation task and improved the LSTM results of [4]. They attributed the success of their model to the hierarchical nature of their network which, when combined with gated linear units and residual connections, allowed them to better capture the structure of the sentences.

Our paper builds primarily off the work of [1] who perform the context-to-text task on the Wikipedia biography dataset through use of a table based context architecture. These contexts allow them to incorporate the various fields of the Wikipedia infobox associated with each word. This context embedding method is further discussed in the following section. However, [1] focus on a simple feed forward neural network with attentive weighting. We will expand upon their approach by incorporating these embeddings into a convolutional encoder-decoder framework with attentive weighting similar to the structures used in [2] and [3].

In our results section, we will compare our models to the baseline from [1] which is a Kneser-Ney model with field-tag embeddings as well as their more advanced NLM with the full contextual embeddings.

3 Methods

In this section we cover the contextual embeddings that were implemented in several of the models attempted. Afterwards we discuss the three models that were attempted to address this problem, a simple LSTM with basic tag embeddings, an LSTM model with advanced contextual embeddings as employed by [1], and finally an encoder-decoder model with a convolutional architecture and self-attentive weighting.

3.1 Embedding Layers

Here we follow the context embedding structure laid out by [1]. They break the infobox contents down into five separate embeddings based on their contents. These include the word, field, starting position, ending position, global field, and global word. These embeddings are concatenated for each token. All tokens that do not have contextual information are mapped to an empty context embedding.

The word embeddings operate similar to your standard embedding leveraging a parameter matrix $\mathbf{E} \in \mathbf{R}^{V \times d}$ where d is the embedding dimension. The field embedding indicates which category of the infobox the word belongs in such as *name, occupation, sports team,* etc. The global conditioning uses the embedding matrices $\mathbf{G^f} \in \mathbf{R}^{F \times g}$ and $\mathbf{G^w} \in \mathbf{R}^{V \times g}$, which map infobox fields and infobox words to an embedding of size g where F is the number of fields in the infobox. These give context of the type of sentence to write for each biography.

There are also positional mappings which give the position of the word in the infobox field relative to the other words in the field. For instance, for an individual with a first, middle and last name in the *name* field, the first name would be in position (1,3), the middle name would be in position (2,2) and the last name is in position (3,1). This gives the model a sense of how many words are in the field and when it should stop writing out names. Each positional indicator is passed through an embedding matrix of size $\mathbf{Z} = \{Z^+, Z^-\} \in \mathbf{R}^{F \times l \times d}$, where l represents the maximum number of words in a sequence. In our implementation, we have $Z^+, Z^- \in \mathbf{R}^{(F)(l) \times d}$ matrices to simplify the embedding of the start and end field-position contexts.

Further details of the embedding structure is discussed in detail in [1], we shall only discuss that each of these elements is mapped to its own embedding to give the model a sense of which words, fields, and field positions are being included. Also, the global field and global word embeddings incorporate embeddings local to the infobox and help provide context as to what the biography is about. For instance, a biography about politicians will have different fields and words than one about athletes.

When training our RNN model with full embedding, the inputs are as follows:

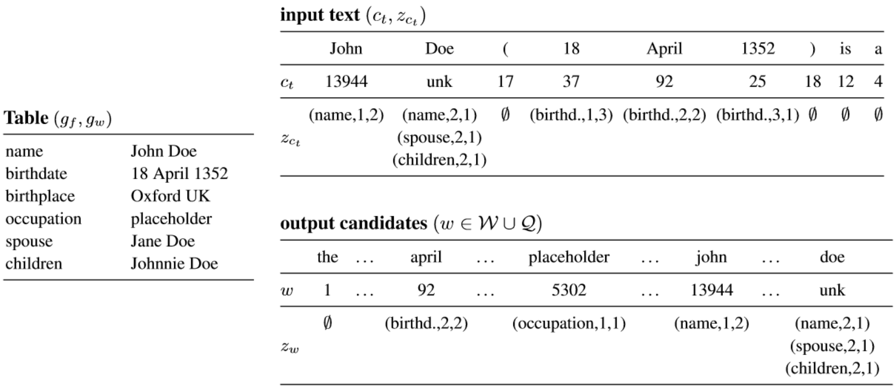


Table (g_f, g_w)

name	John Doe
birthdate	18 April 1352
birthplace	Oxford UK
occupation	placeholder
spouse	Jane Doe
children	Johnnie Doe

input text (c_t, z_{c_t})

	John	Doe	(	18	April	1352	)	is	a
c_t	13944	unk	17	37	92	25	18	12	4
z_{c_t}	(name,1,2)	(name,2,1) (spouse,2,1) (children,2,1)	∅	(birthd.,1,3)	(birthd.,2,2)	(birthd.,3,1)	∅	∅	∅

output candidates $(w \in \mathcal{W} \cup \mathcal{Q})$

	the	…	april	…	placeholder	…	john	…	doe
w	1	…	92	…	5302	…	13944	…	unk
z_w	∅		(birthd.,2,2)		(occupation,1,1)		(name,1,2)		(name,2,1) (spouse,2,1) (children,2,1)

Figure 2: Embedding structure from Lebert 2016. Table features (right) for an example table (left)

Input tensor	Shape
w	$(\cdot, \text{max\_steps})$
z^+	$(\cdot, \text{max\_steps}, Z_{max})$
z^-	$(\cdot, \text{max\_steps}, Z_{max})$
g_f	$(\cdot, \text{max\_steps}, g_{f_{max}})$
g_w	$(\cdot, \text{max\_steps}, g_{w_{max}})$

Where the first dimension takes on the number of elements in the batch, and:

- Z_{max} is the maximum number of times any word from a reference sentence appears in its associated infobox for any reference sentence in the batch,
- $g_{f_{max}}$ is the maximum number of fields of any infobox in any reference sentence in the batch,
- $g_{w_{max}}$ is the maximum number of distinct words of any infobox in any reference sentence in the batch.

Each of the three values above depend on the examples contained within each batch. In each case, elements along the final axis give row positions to look up in the embedding matrix. Upon lookup, the final context embedding is formed by taking the elementwise max across candidate context embeddings and concatenating the results. Further, elements of z^+ and z^- are encoded as (field_id × max_embed_time + field_position), where max_embed_time is the maximum position index fed to the model for a local context word. For instance, if max_embed_time = 10 and word i occurs as the third word of a 15-element field j, then $z_i^+ = (10j + 2)$ and $z_i^- = (10j + 9)$.

3.2 LSTM Baseline Model

Our baseline model is an LSTM structure similar to the basic model that was created for assignment 4. The most basic structure is to simply replace the words in the article with an encoding denoting the field and position that is represented by that word. For instance, in the example from Figure 1, the first sentence *"Joseph Dorville Walter (16 Aug 1895 - 23 May 1995) was a professional footballer who played for Bristol Rovers, Huddersfield Town, Tuanton United and Bath City."* would be tokenized as "`name_1 name_2 name_3 ( birthday_1 birthday_2 birthday_3 - deathdate_1 deathdate_2 deathdate_3 )` *was a professional footballer who played for* `team_3, team_4, team_5` *and* `team_8 .`" This gives us a very simple baseline of what an LSTM can do, but does not factor in contextual information.

For training, the LSTM is fed the start-of-sentence token `<s>` and then generates a best guess at the next word until it eventually generates an end-of-sentence `</s>` token. Hypothesis sentences are generated in a similar manner and are assessed compared to the reference sentences via the BLEU score.

3.3 LSTM Context Embeddings

To improve upon this, we have set up the same model to accept the contextual embeddings as outlined in [1]. This type of embedding should allow the network to learn the context and subject of the article and understand the types of fields that should be included in the final sentence. We expect this model to perform significantly better than the vanilla LSTM due to the additional contextual information incorporated into the model.

We preprocessed word tokens before feeding tokens to this model to drastically reduce the size of our 400K word vocabulary to 50K. We converted URLs, images, commas, periods, round brackets, and URLs to special markers. We stripped all other punctuation from the input.

3.4 Convolutional Seq-to-Seq

For the architecture, we use a sequence-to-sequence structure with attentive weighting cells to connect the encoder and decoder layers. These model architectures have proven successful in a variety of text generation and machine translation tasks and can generate an arbitrarily long sequence of text in spite of the seemingly restrictive nature of the convolutional architecture.

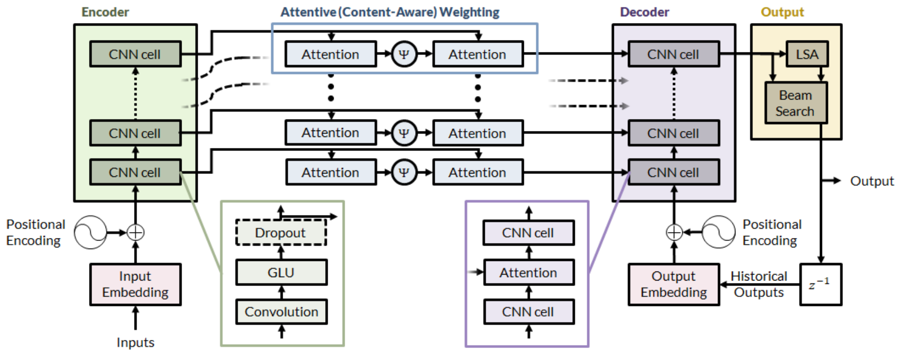


Figure 3: Architecture of Convolutional Sequence-to-Sequence model with neural attentive connection layers in between.

The model has encoder and decoder arrays bridged by attentive an attentive weighting layer, as illustrated by Figure 3.

3.4.1 Convolution Layer

Unlike its more popular use in computer vision, the convolution layer for language model is 1-dimensional along the time axis as implemented by Dauphin et al. in [5] and Gehring et al. in [2]. This is due to the locality assumption not being applicable across the embedding dimension.

The 1-dimensional convolution allows the model to capture the k-gram context on each layer of kernel size k. Cascaded convolution layers allow the model to widen the captured context, and having λ cascaded convolutional layer allows capture of $(\lambda(k-1)+1)$-gram context. We note that large kernel sizes are undesirable due to the need of pre-padding the first token inputs, which will introduce too much padding effect to the training.

There are dropout layers before every convolution layer except for the first. Dropout layers regularize the network training to reduce overfit phenomena. We also include residual connections to boost training speed of the deep architecture.

3.4.2 Gated Linear Unit (GLU)

This unit is devised to allow gating mechanism popularly used in LSTM, adapted by [5] to suit convolutional paradigm. It gates the first half of convolution layer output based on the second half of itself, producing output vector of half the size of convolution layer output.

To deploy this mechanism while enabling convolution cascading of deep architecture, we use filter count of double the input feature size, and use output of half the filters to gate the output of the second half. Dauphin et al. [5] expressed this as two separate kernels, which is equivalent.

3.4.3 Neural Attentive Weighting Layer

In the encoder-decoder model, we allow the outputs of each encoder convolution layer to act as input influence for the decoder convolution layers. We allow each decoder convolution layer to receive independently weighted influence using self-attention mechanism.

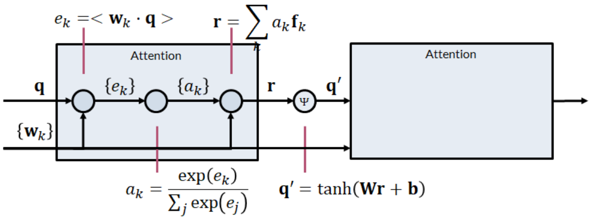

$e_k = < \mathbf{w}_k \cdot \mathbf{q} >$
$\mathbf{r} = \sum_k a_k \mathbf{f}_k$
Attention
$\mathbf{q}$
$\{e_k\}$
$\{a_k\}$
$\{\mathbf{w}_k\}$
$\mathbf{r}$
Ψ
$\mathbf{q}'$
Attention
$a_k = \frac{\exp(e_k)}{\sum_j \exp(e_j)}$
$\mathbf{q}' = \tanh(\mathbf{W}\mathbf{r} + \mathbf{b})$

Figure 4: Self-attention layer from Yang et al. [3]

This self-attention model is much more versatile and allows for high-fidelity aggregation in the computer vision applications. The challenge of implementation comes from selecting the appropriate prior attention, shown as $\mathbf{q}$ in Figure 4. In computer vision, the average network activation shows promising performance as universal prior in [3] and [6]. There are many potential equivalents of this design in NLP application, one of which is average word embedding or convolution layer output activation. We opt to implement this, but acknowledge that this is an open question worth substantial auxiliary research. We discuss other approaches in further works section.

Self-attention layer as implemented by [3], in this NLP application chooses a point in the convex hull spanned by the feature vectors $\{\mathbf{w}_k\}$ along *time* dimension to down-weight less relevant portions of the context.

3.4.4 Neural Attentive Weighted Convolution Padding

In computer vision applications zero-padding input is more acceptable because the input feature space is typically large and the convolution is done in higher dimension. Consequently, the effect of zero-padding is attenuated as they are located at the edges.

We argue that, for NLP applications, convolution layers must be designed percipiently in its behavior of padding the input tensors. Therefore, we propose the concept of padding with self-attention signal. Using the attention bridges, each encoder convolution layer output, through self-attention mechanism, can pad the decoder convolution layers such that zero-padding is avoided while still enabling deep architecture.

3.4.5 Beam Search and LSA

Unlike recursive networks, convolution networks are less straightforward in its expression of transition probability. For this reason, outputs without beam search or standard beam search will gravitate towards repetitive words that minimize logit-based loss but produce unnatural sentences. On the other hand, implementing BLEU-based loss in convolution network will render its parallelism benefits moot, in addition to the complexity of designing a smooth gradient for BLEU that is a discrete and therefore discontinuous metric.

We propose a beam search scheme where the network is allowed to use a training sentence sample as its reference for BLEU-based beam search. BLEU is actually a very suitable metric for beam search as dynamic programming can be exploited in the incremental evaluation of hypotheses. At the time of writing, the dynamic programming implementation is not available yet, but is in progress.

At training time, we use truncated SVD [7] to compute t-dimensional loadings of training table embedding and compute its low-dimensional latent semantic approximation. At search time, we use the computed loadings to project the input table embedding matrix to the t-dimensional space, and find its approximate cosine nearest neighbors in the training set. We then use the nearest neighbors' corresponding training sentences (labels) as the reference of our BLEU-based beam search. Along with copy action [1], this ensemble can produce highly natural sentences as shown in Appendix C very quickly due to low-dimensional projection and matrix computation acceleration available in common modern GPUs.

As a side note, we also considered several approximate nearest neighbor algorithms such as HNSW[8], FAISS[9], and other popular algorithms but at the time of writing, they are either in an unstable state of development or does not support the scale of our dataset.

4 Experimental Results

Due to time constraints all three models were trained using slightly different embedding structures as the more complex contextual embedding took a long time to fully implement. Despite this, we still believe that we can assess the models for their effectiveness of the task as we see clear performance gains from one model to the next.

We also compare our scores to several of the original models employed by [1] to better understand how our methods compare to those of the literature. As previously mentioned, we will compare our model to the Lebret Kneser-Ney baseline model as well as the more advanced NLM with contextual embeddings.

Table 2: Experimental Results

Model	ppx	BLEU
Lebret [1] Kneser-Ney	10.5	2.2
Lebret [1] NLM local context	4.6	26.6
Lebret [1] NLM full context	4.4	34.7
LSTM - tag-embedding	17.4	2.2
LSTM - full-context	-	5.5
CNN Seq-to-Seq	1.1	18.8

4.1 LSTM Baseline Model

As expected the LSTM with the field-tag embedding structure performed the worst. The BLEU score is 2.2 which is similar to the Kneser-Ney from [1]. As can be seen in the Appendix A the model does typically get the first name correct and sometimes attempts to generate dates for birth and death, but the sentences are not very intelligible and only occasionally are related to the profession or topic of the individual. This model would likely benefit from beam search and more training time.

4.2 LSTM - Context Embeddings

Providing context inputs z^+, z^- allowed this model slightly better performance than the baseline model on our held-out test data, achieving a BLEU of 5.5. However, the new embeddings also increase the number of parameters in to the model, making it difficult to determine whether the increased score is due to the emeddings imparting meaningful information, or if the increased score is simply the result of increased model size. We used a train/val/test split of 80/10/10 over 582,659 examples, a vocab size of 50,000, and hidden dimension of 100, giving the 5 embedding matrices as: $\mathbf{W} \in \mathbf{R}^{50000\times100}$, $\mathbf{Z}^+ \in \mathbf{R}^{(10*3873)\times100}$, $\mathbf{Z}^- \in \mathbf{R}^{(10*3873)\times100}$, $\mathbf{G^f} \in \mathbf{R}^{3873\times100}$, and $\mathbf{G^w} = \mathbf{W}$ (per [1]).

4.3 Convolutional Seq-to-Seq

The sequence-to-sequence CNN outperformed both of our other models by a wide margin. The BLEU score of 18.8 was far better than our baseline model. This is likely due to a variety of factors. The added attention allows for the context of the infobox to passed from the encoder to the decoder allowing for longer and richer memory. The LSA beam search also allowed us to assess a variety of candidate sentences and choosing the one that would be the best fit for our data.

4.3.1 Error Analysis

From the example sentences in the Appendix C we can see that this model performs far better and makes much more reasonable sentences than the other models. The example sentences in this case are ordered by BLEU score from highest to lowest. There are several sentences for which it performs exceptionally well. Also, most of the sentences are readable and have clean grammatical structures, which is obviously a good quality to have when trying to generate article sentences.

There are several cases that display some obvious flaws with the model. In the poorest performing example, the model simply terminates after the first two words of the sentence, printing only *"charles e."* before the end-of-sentence token. This was a somewhat common problem that we experienced while training this model. This is possibly due to the period at the end of the abbreviated name. This may be an artifact of the training data, as sentences in the training data may have been prematurely terminated due to abbreviations. While it is important to validate that the training data is correct, it is also necessary for our model to have the capability of writing abbreviations consistently as they are a common feature of these biographies.

Another example that performed poorly was a name which incorporated Chinese characters and a detailed pronunciation of a name. It is understandable that the model would struggle with these sentences as the tokens are very uncommon.

While we are very pleased with the performance of the CNN architecture, the BLEU score of 18.8 fell short of the benchmark score of 34.7 from Lebret's NLM with the full context embedding as well as the 26.6 from the NLM with local context embeddings. The poor performance in comparison to the local embeddings could be a result of the fact that we were not able to fully explore the hyperparameter space due to the limited time constraints. We believe that with more time to train the model and tune the parameters, we could get this method very close to the 26.6 for the local context embeddings.

5 Conclusion & Next Steps

Based on our results we can clearly state two conclusions. First, that the contextual embedding methodology developed by [1] is a very effective way to encode the context of the infobox data and it dramatically out-performs that of the basic tag-embeddings. Specifically, we concur that the copy action is the primary means to achieving reasonable performance. Secondly, it's clear that the CNN sequence-to-sequence framework is a far superior model. These results are obviously not surprising given the recent performance of attention in similar tasks.

To further our work, we would obviously like to explore implementing the full contextual embedding for the CNN architecture. For [1] this resulted in an 8-point increase in BLEU and seems the first logical step. Additionally, having more time to tune could dramatically improve the overall output as well as exploring methods for handling rare tokens and addressing the early stopping issue.

Another consideration point is using pre-trained embedding as have been shown by [10] to aid performance. It was previously considered as part of current paper's work, however the issue of imputing table field name tokens as part of copy action method presented a question of how to fairly impute the embeddings of these tokens. A suggestion would be to consider the Bayesian expected value of the embeddings.

We strongly concur with [1] in their call for a more appropriate metric for sentence generation. While BLEU's n-gram approach is acceptable for modest-performance machines, it is inflexible in the way of synonyms and higher level sentence naturalness. For instance, in Example 3 of Appendix C, "judo player" may be an acceptable synonym of "judoka".

Other simple fixes could have improved performance as well. They were not yet implemented due to time constraint, but is a very useful note for further works:

- Copy action should be improved by rule-based substitution of generated fields that are not part of the input fields (fixes e.g. `[clubs_1]` and `[occupation_1]` tokens in Appendix C).
- Incorporating POS-based likelihood as beam search probability signal will help produce more natural sentences.
- Output from affine layer of decoder convolution was used, but we experimented with a version where cascaded attention is then applied to the output of the affine layer. It generated repetitive tokens, but this could have been pursued further with the help of pointer-generator network

Acknowledgement

We would like to thank Hao Ma from Microsoft Research who have summarized a few recent advances in this topic and inspired us to research further on it. We would also like to thank James Kunz, Nithum Thain, Ian Tenney, Daniel Cer, and Joachim Rahmfeld for their helpful feedback and consistent support as instructors and teaching assistant for Natural Language Processing with Deep Learning at University of California, Berkeley.

References

[1] Rémi Lebret, David Grangier, and Michael Auli. Neural text generation from structured data with application to the biography domain. *arXiv preprint arXiv:1603.07771*, 2016.

[2] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*, 2017.

[3] Jiaolong Yang, Peiran Ren, Dong Chen, Fang Wen, Hongdong Li, and Gang Hua. Neural aggregation network for video face recognition. *arXiv preprint arXiv:1603.05474*, 2016.

[4] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[5] Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. *CoRR*, abs/1612.08083, 2016.

[6] P. A. B. Hendri. draft. unpublished.

[7] N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *ArXiv e-prints*, September 2009.

[8] Yury A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *CoRR*, abs/1603.09320, 2016.

[9] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*, 2017.

[10] Rémi Lebret and Ronan Collobert. Rehabilitation of count-based models for word vector representations. *CoRR*, abs/1412.4930, 2014.

[11] Hongyuan Mei, Mohit Bansal, and Matthew R Walter. What to talk about and how? selective generation using lstms with coarse-to-fine alignment. *arXiv preprint arXiv:1509.00838*, 2015.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

[13] K. Fan. draft. unpublished.

[14] Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics, 2003.

[15] Daniel Cer, Daniel Jurafsky, and Christopher D Manning. Regularization and search for minimum error rate training. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 26–34. Association for Computational Linguistics, 2008.

[16] Anja Belz and Ehud Reiter. Comparing automatic and human evaluation of nlg systems. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, 2006.

Appendix A Example sentences for LSTM with tag-embeddings

Example 1	Article	https://en.wikipedia.org/wiki/Peter_Gawthorne
	Actual	Peter Gawthorne (1 September 1884 – 17 March 1962) was an Anglo-Irish actor, probably best known for his roles in Will Hay films.
	Generated	peter <unk>) from the russian fourth national league football right-handed defensive linebacker and known to her career for both.
Example 2	Article	https://en.wikipedia.org/wiki/G._Cole_(1811_cricketer)
	Actual	G. Cole (first name and dates unknown) was an English first-class cricketer associated with Marylebone Cricket Club (MCC) who was active in the 1810s.
	Generated	g. cole (died write as the current australian baseball football league.
Example 3	Article	https://en.wikipedia.org/wiki/Alexander_Wiley
	Actual	Alexander Wiley (May 26, 1884 – October 26, 1967) was a Republican who served four terms in the United States Senate for the state of Wisconsin from 1939 to 1963.
	Generated	alexander wiley [name_3] [name_4] ((b.) (; 1992 – 26 may 26 may 1884) 1884)
Example 4	Article	https://en.wikipedia.org/wiki/Dennis_Grainger
	Actual	Dennis Grainger (5 March 1920 – 6 June 1986) was an English professional footballer who played as a left winger.
	Generated	dennis grainger is a sports activist is an american one of [currentclub_1] [currentclub_2], <unk> [occupation_7] of american group under only concentration centuries.

Appendix B Example sentences for LSTM with context-embeddings

Example 1	Article	https://en.wikipedia.org/wiki/Anna_Klingmann
	Actual	Anna Klingmann (born 1965) is an architect and academic who specializes in branding.
	Generated	Anna Klingmann [birth_place_3] Germany s [known_for_5] [occupation_1] at UC .
Example 2	Article	https://en.wikipedia.org/wiki/Erik_Tammer)
	Actual	Erik Tammer (born June 29, 1969 in Utrecht) is a former professional association football striker from the Netherlands.
	Generated	Erik Tammer owner player who played n-nas-ss Erik [article_title_8] Tammer is <unk> Tammer republic is a footballer.
Example 3	Article	https://en.wikipedia.org/wiki/Pierre_Offerman
	Actual	Pierre Offerman was the commandant of the <unk> Elephant <unk> Center and chief warden of the Conservation Service of the Belgian Congo, ...
	Generated	Pierre Offerman are a engineer best <PAD> <PAD> <PAD> ...
Example 4	Article	https://en.wikipedia.org/wiki/Brockway_McMillan
	Actual	Brockway McMillan (born March 30, 1915) is a retired American government official and scientist, who served as the eighth ...
	Generated	she professional [party_2] [birth_place_4] the 30 March then – [death_date_2].

Appendix C Example sentences for CNN encoder-decoder

Example 1	Article	https://en.wikipedia.org/wiki/Paul_Brooks_(cricketer)
	Actual	paul wilson brooks -lrb- 28 may 1921 – 26 january 1946 -rrb- was an english cricketer .
	Generated	paul wilson brooks -lrb- 28 may 1921 – 26 january 1946 -rrb- was an english cricketer .
Example 2	Article	https://en.wikipedia.org/wiki/Jack_Adamson
	Actual	john h. “ jack ” adamson -lrb- 6 january 1873 – 2 october 1937 -rrb- was an australian rules footballer who played three games for south melbourne in the victorian football league between 1897 and 1898 .
	Generated	john h. “ jack ” adamson -lrb- 6 january 1873 – 2 october 1937 -rrb- was an australian rules footballer who played for the [clubs_1] football club melbourne and was a part of the bombers ’ 1897 premiership team .
Example 3	Article	https://en.wikipedia.org/wiki/Mar%C3%ADa_Villapol
	Actual	maría elena villapol blanca -lrb- born november 16 , 1967 -rrb- is a retired female judoka from venezuela .
	Generated	maría villapol -lrb- born november 16 , 1967 in friesland -rrb- is a female judo player from the netherlands , who plays in different positions
Example 4	Article	https://en.wikipedia.org/wiki/Gregory_Tarver
	Actual	gregory williams tarver , sr. , known as greg tarver -lrb- born march 30 , 1946 -rrb- , is an african american businessman and democratic politician in shreveport , louisiana , who served on the shreveport city council from 1978 to 1984 and as a louisiana state senator from the predominantly black district 39 in caddo parish from 1984 to 2004 .
	Generated	gregory williams “ greg ” tarver , sr. , -lrb- born march 30 , 1946 -rrb- is a democratic former member of the louisiana state senate from shreveport , a small community south of alexandria , louisiana , the seat of government of caddo parish and the largest city in the central louisiana region .
Example 5	Article	https://en.wikipedia.org/wiki/Margaret_E._Barber
	Actual	margaret emma barber or m. e. barber -lrb- 1866 – 1930 ; chinese ; pinyin : “ hé shòuēn ” ; foochow romanized : “ huò sêu-ŏng ” -rrb- , was a british missionary in china .
	Generated	margaret e. barber -lrb- 1866 – 1929 -rrb- was a prominent [occupation_1] and civic leader in roc suffolk , in the early 20th century .
Example 6	Article	https://en.wikipedia.org/wiki/Charles_E._Rosenberg
	Actual	charles e. rosenberg -lrb- born november 11 , 1936 -rrb- is an american historian of medicine , he is professor of the history of science and the ernest e. monrad professor in the social sciences at harvard university .
	Generated	charles e.