Entity Commonsense Representation for Neural Abstractive Summarization

Reinald Kim Amplayo* and **Seonjae Lim*** and **Seung-won Hwang**
Yonsei University, Seoul, South Korea
{rktamplayo, sun.lim, seungwonh}@yonsei.ac.kr

Abstract

A major proportion of a text summary includes important entities found in the original text. These entities build up the topic of the summary. Moreover, they hold commonsense information once they are linked to a knowledge base. Based on these observations, this paper investigates the usage of linked entities to guide the decoder of a neural text summarizer to generate concise and better summaries. To this end, we leverage on an off-the-shelf entity linking system (ELS) to extract linked entities and propose **Entity2Topic (E2T)**, a module easily attachable to a sequence-to-sequence model that transforms a list of entities into a vector representation of the topic of the summary. Current available ELS's are still not sufficiently effective, possibly introducing unresolved ambiguities and irrelevant entities. We resolve the imperfections of the ELS by (a) encoding entities with selective disambiguation, and (b) pooling entity vectors using firm attention. By applying E2T to a simple sequence-to-sequence model with attention mechanism as base model, we see significant improvements of the performance in the Gigaword (sentence to title) and CNN (long document to multi-sentence highlights) summarization datasets by at least 2 ROUGE points.

1 Introduction

Text summarization is a task to generate a shorter and concise version of a text while preserving the meaning of the original text. The task can be divided into two subtask based on the approach: extractive and abstractive summarization. Extractive summarization is a task to create summaries by pulling out snippets of text form the original text and combining them to form a summary. Abstractive summarization asks to generate summaries from scratch without the restriction to use the available words from the original text. Due to the limitations of extractive summarization on incoherent texts and unnatural methodology (Yao et al., 2017), the research trend has shifted towards abstractive summarization.

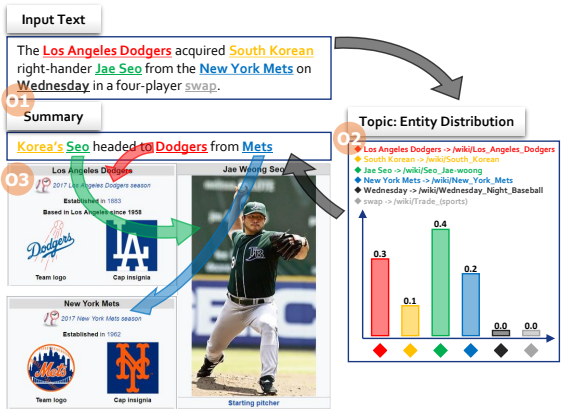

Figure 1: Observations on linked entities in summaries. **O1**: Summaries are mainly composed of entities. **O2**: Entities can be used to represent the topic of the summary. **O3**: Entity commonsense learned from a large corpus can be used.

Sequence-to-sequence models (Sutskever et al., 2014) with attention mechanism (Bahdanau et al., 2014) have found great success in generating abstractive summaries, both from a single sentence (Chopra et al., 2016) and from a long document with multiple sentences (Chen et al., 2016). However, when generating summaries, it is necessary to determine the main topic and to sift out unnecessary information that can be omitted. Sequence-to-sequence models have the tendency to include all the information, relevant or not, that are found in the original text. This may result to unconcise summaries that concentrates wrongly on irrelevant topics. The problem is especially severe when summarizing longer texts.

In this paper, we propose to use entities found in

* Amplayo and Lim are co-first authors with equal contribution. Names are arranged alphabetically.

the original text to infer the summary topic, mitigating the aforementioned problem. Specifically, we leverage on linked entities extracted by employing a readily available entity linking system. The importance of using linked entities in summarization is intuitive and can be explained by looking at Figure 1 as an example. First (**O1** in the Figure), aside from auxiliary words to construct a sentence, a summary is mainly composed of linked entities extracted from the original text. Second (**O2**), we can depict the main topic of the summary as a probability distribution of relevant entities from the list of entities. Finally (**O3**), we can leverage on entity commonsense learned from a separate large knowledge base such as Wikipedia.

To this end, we present a method to effectively apply linked entities in sequence-to-sequence models, called **Entity2Topic (E2T)**. E2T is a module that can be easily attached to any sequence-to-sequence based summarization model. The module encodes the entities extracted from the original text by an entity linking system (ELS), constructs a vector representing the topic of the summary to be generated, and informs the decoder about the constructed topic vector. Due to the imperfections of current ELS's, the extracted linked entities may be too **ambiguous** and **coarse** to be considered relevant to the summary. We solve this issue by using entity encoders with **selective disambiguation** and by constructing topic vectors using **firm attention**.

We experiment on two datasets, Gigaword and CNN, with varying lengths. We show that applying our module to a sequence-to-sequence model with attention mechanism significantly increases its performance on both datasets. Moreover, when compared with the state-of-the-art models for each dataset, the model obtains a comparable performance on the Gigaword dataset where the texts are short, and outperforms all competing models on the CNN dataset where the texts are longer. Furthermore, we provide analysis on how our model effectively uses the extracted linked entities to produce concise and better summaries.

2 Usefulness of linked entities in summarization

In the next subsections, we present detailed arguments with empirical and previously examined evidences on the observations and possible issues when using linked entities extracted by an entity linking system (ELS) for generating abstractive summaries. For this purpose, we use the development sets of the Gigaword dataset provided in (Rush et al., 2015) and of the CNN dataset provided in (Hermann et al., 2015) as the experimental data for quantitative evidence and refer the readers to Figure 1 as the running example.

2.1 Observations

As discussed in Section 1, we find three observations that show the usefulness of linked entities for abstractive summarization.

First, summaries are mainly composed of linked entities extracted from the original text. In the example, it can be seen that the summary contains four words that refer to different entities. In fact, all noun phrases in the summary mention at least one linked entity. In our experimental data, we extract linked entities from the original text and compare them to the noun phrases found in the summary. We report that 77.1% and 75.1% of the noun phrases on the Gigaword and CNN datasets, respectively, contain at least one linked entity, which confirms our observation.

Second, linked entities can be used to represent the topic of the summary, defined as a multinomial distribution over entities, as graphically shown in the example, where the probabilities refer to the relevance of the entities. Entities have been previously used to represent topics (Newman et al., 2006), as they can be utilized as a controlled vocabulary of the main topics in a document (Hulpus et al., 2013). In the example, we see that the entity "*Jae Seo*" is the most relevant because it is the subject of the summary, while the entity "*South Korean*" is less relevant because it is less important when constructing the summary.

Third, we can make use of the entity commonsense that can be learned as a continuous vector representation from a separate larger corpus (Ni et al., 2016; Yamada et al., 2017). In the example, if we know that the entities "*Los Angeles Dodgers*" and "*New York Mets*" are American baseball teams and "*Jae Seo*" is a baseball player associated with the teams, then we can use this information to generate more coherent summaries. We find that 76.0% of the extracted linked entities are covered by the pre-trained vectors[1] in our experimental data, proving our third observation.

[1] https://github.com/idio/wiki2vec

2.2 Possible issues

Despite its usefulness, linked entities extracted from ELS's have issues because of low precision rates (Hasibi et al., 2016) and design challenges in training datasets (Ling et al., 2015). These issues can be summarized into two parts: ambiguity and coarseness.

First, the extracted entities may be ambiguous. In the example, the entity "*South Korean*" is ambiguous because it can refer to both the South Korean person and the South Korean language, among others[2]. In our experimental data, we extract (1) the top 100 entities based on frequency, and (2) the entities extracted from 100 randomly selected texts, and check whether they have disambiguation pages in Wikipedia or not. We discover that 71.0% of the top 100 entities and 53.6% of the entities picked at random have disambiguation pages, which shows that most entities are prone to ambiguity problems.

Second, the linked entities may also be too common to be considered an entity. This may introduce *errors* and *irrelevance* to the summary. In the example, "*Wednesday*" is erroneous because it is wrongly linked to the entity "*Wednesday Night Baseball*". Also, "*swap*" is irrelevant because although it is linked correctly to the entity "*Trade (Sports)*", it is too common and irrelevant when generating the summaries. In our experimental data, we randomly select 100 data instances and tag the correctness and relevance of extracted entities into one of four labels: A: correct and relevant, B: correct and somewhat relevant, C: correct but irrelevant, and D: incorrect. Results show that 29.4%, 13.7%, 30.0%, and 26.9% are tagged with A, B, C, and D, respectively, which shows that there is a large amount of incorrect and irrelevant entities.

3 Our model

To solve the issues described above, we present **Entity2Topic (E2T)**, a module that can be easily attached to any sequence-to-sequence based abstractive summarization model. E2T encodes the linked entities extracted from the text and transforms them into a single topic vector. This vector is ultimately concatenated to the decoder hidden state vectors. The module contains two submodules specifically for the issues presented by the entity linking systems: the entity encoding submodule with selective disambiguation and the pooling submodule with firm attention.

Overall, our full architecture can be illustrated as in Figure 2, which consists of an entity linking system (ELS), a sequence-to-sequence with attention mechanism model, and the E2T module. We note that our proposed module can be easily attached to more sophisticated abstractive summarization models (Zhou et al., 2017; Tan et al., 2017) that are based on the traditional encoder-decoder framework and consequently can produce better results. The code of the base model and the E2T are available online[3].

3.1 Base model

As our base model, we employ a basic encoder-decoder RNN used in most neural machine translation (Bahdanau et al., 2014) and text summarization (Nallapati et al., 2016) tasks. We employ a two-layer bidirectional GRU (BiGRU) as the recurrent unit of the encoder. The BiGRU consists of a forward and backward GRU, which results to sequences of forward and backward hidden states $(\overrightarrow{h}_1, \overrightarrow{h}_2, ..., \overrightarrow{h}_n)$ and $(\overleftarrow{h}_1, \overleftarrow{h}_2, ..., \overleftarrow{h}_n)$, respectively:

$$\overrightarrow{h}_i = GRU(x_i, \overrightarrow{h}_{i-1})$$
$$\overleftarrow{h}_i = GRU(x_i, \overleftarrow{h}_{i+1})$$

The forward and backward hidden states are concatenated to get the hidden state vectors of the tokens (i.e. $h_i = [\overrightarrow{h}_i; \overleftarrow{h}_i]$). The final states of the forward and backward GRU are also concatenated to create the final text representation vector of the encoder $s = [\overrightarrow{h}_n; \overleftarrow{h}_1]$. These values are calculated per layer, where x_t of the second layer is h_t of the first layer. The final text representation vectors are projected by a fully connected layer and are passed to the decoder as the initial hidden states $s_0 = s$.

For the decoder, we use a two-layer unidirectional GRU with attention. At each time step t, the previous token y_{t-1}, the previous hidden state s_{t-1}, and the previous context vector c_{t-1} are passed to a GRU to calculate the new hidden state s_t, as shown in the equation below.

$$s_t = GRU(w_{t-1}, s_{t-1}, c_{t-1})$$

[2] https://en.wikipedia.org/wiki/South_Korean

[3] https://github.com/rktamplayo/Entity2Topic

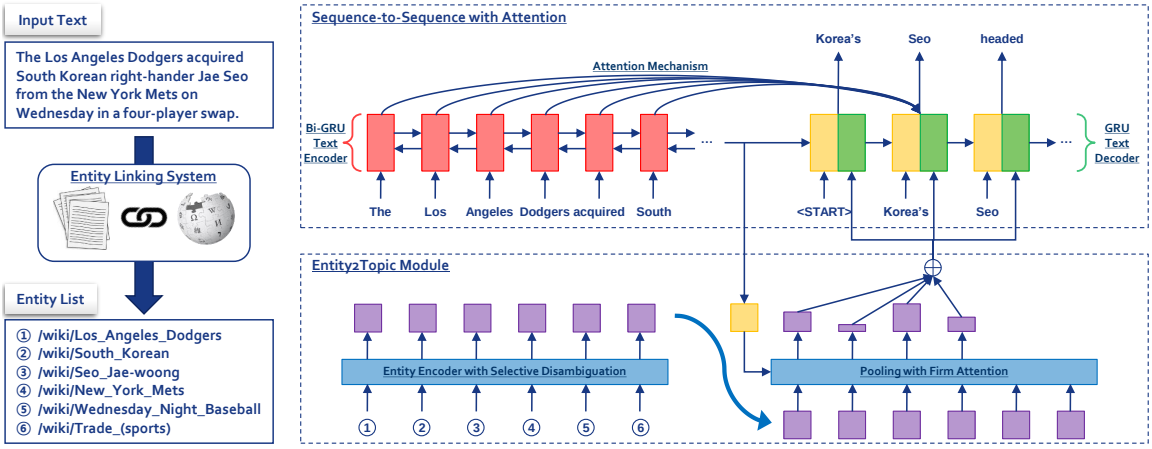

Figure 2: Full architecture of our proposed sequence-to-sequence model with Entity2Topic (E2T) module.

The context vector c_t is computed using the additive attention mechanism (Bahdanau et al., 2014), which matches the current decoder state s_t and each encoder state h_i to get an importance score. The scores are then passed to a softmax and are used to pool the encoder states using weighted sum. The final pooled vector is the context vector, as shown in the equations below.

$$g_{t,i} = v_a^\top tanh(W_a s_{t-1} + U_a h_i)$$
$$a_{t,i} = \frac{exp(g_{t,i})}{\sum_i exp(g_{t,i})}$$
$$c_t = \sum_i a_{t,i} h_i$$

Finally, the previous token y_{t-1}, the current context vector c_t, and the current decoder state s_t are used to generate the current word y_t with a softmax layer over the decoder vocabulary, as shown below.

$$o_t = W_w w_{t-1} + W_c c_t + W_s s_t$$
$$p(y_t|y_{<t}) = softmax(W_o o_t)$$

3.2 Entity encoding submodule

After performing entity linking to the input text using the ELS, we receive a sequential list of linked entities, arranged based on their location in the text. We embed these entities to d-dimensional vectors $E = \{e_1, e_2, ..., e_m\}$ where $e_i \in \mathbb{R}^d$. Since these entities may still contain ambiguity, it is necessary to resolve them before applying them to the base model. Based on the idea that an ambiguous entity can be disambiguated using its neighboring entities, we introduce two kinds of disambiguating encoders below.

Globally disambiguating encoder One way to disambiguate an entity is by using all the other entities, putting more importance to entities that are nearer. For this purpose, we employ an RNN-based model to globally disambiguate the entities. Specifically, we use BiGRU and concatenate the forward and backward hidden state vectors as the new entity vector:

$$\overrightarrow{h}_i = GRU(e_i, \overrightarrow{h}_{i-1})$$
$$\overleftarrow{h}_i = GRU(e_i, \overleftarrow{h}_{i+1})$$
$$e'_i = [\overrightarrow{h}_i; \overleftarrow{h}_i]$$

Locally disambiguating encoder Another way to disambiguate an entity is by using only the direct neighbors of the entity, putting no importance value to entities that are far. To do this, we employ a CNN-based model to locally disambiguate the entities. Specifically, we do the convolution operation using filter matrices $W_f \in \mathbb{R}^{h \times d}$ with filter size h to a window of h words. We do this for different sizes of h. This produces new feature vectors $c_{i,h}$ as shown below, where $f(.)$ is a non-linear function:

$$c_{i,h} = f([e_{i-(h-1)/2}; ...; e_{i+h(+1)/2}]^\top W_f + b_f)$$

The convolution operation reduces the number of entities differently depending on the filter size h. To prevent loss of information and to produce the same amount of feature vectors $c_{i,h}$, we pad the entity list dynamically such that when the filter size is h, the number of paddings on each side is $(h-1)/2$. The filter size h therefore refers to the number of entities used to disambiguate a middle entity. Finally, we concatenate all feature vectors

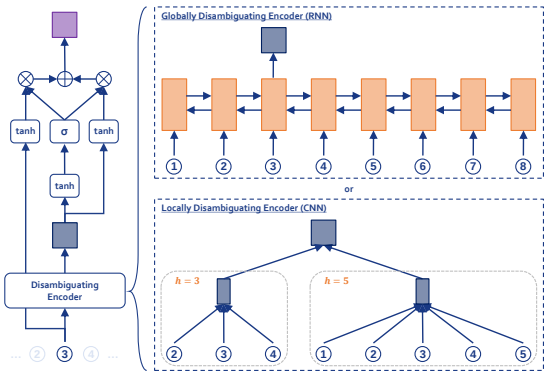

Figure 3: Entity encoding submodule with selective disambiguation applied to the entity ③. The left figure represents the full submodule while the right figure represents the two choices of disambiguating encoders.

of different h's for each i as the new entity vector:

$$e'_i = [c_{i,h_1}; c_{i,h_2}; ...]$$

The question on which disambiguating encoder is better has been a debate; some argued that using only the local context is appropriate (Lau et al., 2013) while some claimed that additionally using global context also helps (Wang et al., 2015). The RNN-based encoder is good as it smartly makes use of all entities, however it may perform bad when there are many entities as it introduces noise when using a far entity during disambiguation. The CNN-based encoder is good as it minimizes the noise by totally ignoring far entities when disambiguating, however determining the appropriate filter sizes h needs engineering. Overall, we argue that when the input text is short (e.g. a sentence), both encoders perform comparably, otherwise when the input text is long (e.g. a document), the CNN-based encoder performs better.

Selective disambiguation It is obvious that not all entities need to be disambiguated. When a correctly linked and already adequately disambiguated entity is disambiguated again, it would make the entity very context-specific and might not be suitable for the summarization task. Our entity encoding submodule therefore uses a selective mechanism that decides whether to use the disambiguating encoder or not. This is done by introducing a selective disambiguation gate d. The final entity vector $\tilde{e}_i$ is calculated as the linear transformation of e_i and e'_i:

$$\begin{aligned} e'_i &= encoder(e_i) \\ d &= \sigma(W_d e'_i + b_d) \\ \tilde{e}_i &= d \times f(W_x e_i + b_x) + \\ &\quad (1-d) \times f(W_y e'_i + b_y) \end{aligned}$$

The full entity encoding submodule is illustrated in Figure 3. Ultimately, the submodule outputs the disambiguated entity vectors $\tilde{E} = \{\tilde{e}_1, \tilde{e}_2, ..., \tilde{e}_m\}$.

3.3 Pooling submodule

The entity vectors $\tilde{E}$ are pooled to create a single topic vector t that represents the topic of the summary. One possible pooling technique is to use soft attention (Xu et al., 2015) on the vectors to determine the importance value of each vector, which can be done by matching each entity vector with the text vector s from the text encoder as the context vector. The entity vectors are then pooled using weighted sum. One problem with soft attention is that it considers *all* entity vectors when constructing the topic vector. However, not all entities are important and necessary when generating summaries. Moreover, a number of these entities may be erroneous and irrelevant, as reported in Section 2.2. Soft attention gives non-negligible important scores to these entities, thus adds unnecessary noise to the construction of the topic vector.

Our pooling submodule instead uses **firm attention** mechanism to consider only top k entities when constructing the topic vector. This is done in a differentiable way as follows:

$$\begin{aligned} G &= v_a^\top tanh(W_a \tilde{E} + U_a s) \\ K &= top\_k(G) \\ P &= sparse\_vector(K, 0, -\infty) \\ g'_i &= g_i + p_i \\ a_i &= \frac{exp(g'_i)}{\sum_i exp(g'_i)} \\ t &= \sum_i a_i \tilde{e}_i \end{aligned}$$

where the functions $K = top\_k(G)$ gets the indices of the top k vectors in G and $P = sparse\_vector(K, 0, -\infty)$ creates a sparse vector where the values of K is 0 and $-\infty$ otherwise[4]. The sparse vector P is added to the original importance score vector G to create a new importance

[4]We use -10^9 to represent $-\infty$.

score vector. In this new vector, important scores of non-top k entities are $-\infty$. When softmax is applied, this gives very small, negligible, and close-to-zero values to non-top k entities. The value k depends on the lengths of the input text and summary. Moreover, when k increases towards infinity, firm attention becomes soft attention. We decide k empirically (see Section 5).

3.4 Extending from the base model

Entity2Topic module extends the base model as follows. The final text representation vector s is used as a context vector when constructing the topic vector t in the pooling submodule. The topic vector t is then concatenated to the decoder hidden state vectors s_i, i.e. $s'_i = [s_i; t]$. The concatenated vector is finally used to create the output vector:

$$o_i = W_w w_{i-1} + W_c c_i + W_s s'_i$$

4 Related work

Due to its recent success, neural network models have been used with competitive results on abstractive summarization. A neural attention model was first applied to the task, easily achieving state-of-the-art performance on multiple datasets (Rush et al., 2015). The model has been extended to instead use recurrent neural network as decoder (Chopra et al., 2016). The model was further extended to use a full RNN encoder-decoder framework and further enhancements through lexical and statistical features (Nallapati et al., 2016). The current state-of-the-art performance is achieved by selectively encoding words as a process of distilling salient information (Zhou et al., 2017).

Neural abstractive summarization models have also been explored to summarize longer documents. Word extraction models have been previously explored, performing worse than sentence extraction models (Cheng and Lapata, 2016). Hierarchical attention-based recurrent neural networks have also been applied to the task, owing to the idea that there are multiple sentences in a document (Nallapati et al., 2016). Finally, distraction-based models were proposed to enable models to traverse the text content and grasp the overall meaning (Chen et al., 2016). The current state-of-the-art performance is achieved by a graph-based attentional neural model, considering the key factors of document summarization such as saliency, fluency and novelty (Tan et al., 2017).

Dataset	Gigaword	CNN
num(data)	4.0M	84K
avg(inputWord)	31.4	774.9
avg(outputWord)	8.2	48.1
min(inputEntity)	1	1
max(inputEntity)	36	743
avg(inputEntity)	4.5	94.6

Table 1: Dataset statistics.

Previous studies on the summarization tasks have only used entities in the preprocessing stage to anonymize the dataset (Nallapati et al., 2016) and to mitigate out-of-vocabulary problems (Tan et al., 2017). Linked entities for summarization are still not properly explored and we are the first to use linked entities to improve the performance of the summarizer.

5 Experimental settings

Datasets We use two widely used summarization datasets with different text lengths. First, we use the Annotated English Gigaword dataset as used in (Rush et al., 2015). This dataset receives the first sentence of a news article as input and use the headline title as the gold standard summary. Since the development dataset is large, we randomly selected 2000 pairs as our development dataset. We use the same held-out test dataset used in (Rush et al., 2015) for comparison. Second, we use the CNN dataset released in (Hermann et al., 2015). This dataset receives the full news article as input and use the human-generated multiple sentence highlight as the gold standard summary. The original dataset has been modified and preprocessed specifically for the document summarization task (Nallapati et al., 2016). In addition to the previously provided datasets, we extract linked entities using Dexter[5] (Ceccarelli et al., 2013), an open source ELS that links text snippets found in a given text to entities contained in Wikipedia. We use the default recommended parameters stated in the website. We summarize the statistics of both datasets in Table 1.

Implementation For both datasets, we further reduce the size of the input, output, and entity vocabularies to at most 50K as suggested in (See et al., 2017) and replace less frequent words to

[5] http://dexter.isti.cnr.it/

"<unk>". We use 300D Glove[6] (Pennington et al., 2014) and 1000D wiki2vec[7] pre-trained vectors to initialize our word and entity vectors. For GRUs, we set the state size to 500. For CNN, we set $h = 3, 4, 5$ with $400, 300, 300$ feature maps, respectively. For firm attention, k is tuned by calculating the perplexity of the model starting with smaller values (i.e. $k = 1, 2, 5, 10, 20, ...$) and stopping when the perplexity of the model becomes worse than the previous model. Our preliminary tuning showed that $k = 5$ for Gigaword dataset and $k = 10$ for CNN dataset are the best choices. We use dropout (Srivastava et al., 2014) on all non-linear connections with a dropout rate of 0.5. We set the batch sizes of Gigaword and CNN datasets to 80 and 10, respectively. Training is done via stochastic gradient descent over shuffled mini-batches with the Adadelta update rule, with l_2 constraint (Hinton et al., 2012) of 3. We perform early stopping using a subset of the given development dataset. We use beam search of size 10 to generate the summary.

Baselines For the Gigaword dataset, we compare our models with the following abstractive baselines: **ABS+** (Rush et al., 2015) is a fine tuned version of ABS which uses an attentive CNN encoder and an NNLM decoder, **Feat2s** (Nallapati et al., 2016) is an RNN sequence-to-sequence model with lexical and statistical features in the encoder, **Luong-NMT** (Luong et al., 2015) is a two-layer LSTM encoder-decoder model, **RAS-Elman** (Chopra et al., 2016) uses an attentive CNN encoder and an Elman RNN decoder, and **SEASS** (Zhou et al., 2017) uses BiGRU encoders and GRU decoders with selective encoding. For the CNN dataset, we compare our models with the following extractive and abstractive baselines: **Lead-3** is a strong baseline that extracts the first three sentences of the document as summary, **LexRank** extracts texts using LexRank (Erkan and Radev, 2004), **Bi-GRU** is a non-hierarchical one-layer sequence-to-sequence abstractive baseline, **Distraction-M3** (Chen et al., 2016) uses a sequence-to-sequence abstractive model with distraction-based networks, and **GBA** (Tan et al., 2017) is a graph-based attentional neural abstractive model. All baseline results used beam search and are gathered from previous papers. Also, we compare our final model **BASE+E2T** with the base model **BASE** and some variants of our model (without selective disambiguation, using soft attention).

[6] https://nlp.stanford.edu/projects/glove/

[7] https://github.com/idio/wiki2vec

Model	RG-1	RG-2	RG-L
BASE: s2s+att	34.14	15.44	32.47
BASE+E2T$_{cnn+sd}$	**37.04**	16.66	**34.93**
BASE+E2T$_{rnn+sd}$	36.89	**16.86**	34.74
BASE+E2T$_{cnn}$	36.56	16.56	34.57
BASE+E2T$_{rnn}$	36.52	16.21	34.32
BASE+E2T$_{cnn+soft}$	36.56	16.44	34.58
BASE+E2T$_{rnn+soft}$	36.38	16.12	34.20
ABS+	29.78	11.89	26.97
Feat2s	32.67	15.59	30.64
Luong-NMT	33.10	14.45	30.71
RAS-Elman	33.78	15.97	31.15
SEASS	**36.15**	**17.54**	**33.63**

Table 2: Results on the Gigaword dataset using the full-length F1 variants of ROUGE.

Model	RG-1	RG-2	RG-L
BASE: s2s+att	25.5	5.8	20.0
BASE+E2T$_{cnn+sd}$	**31.9**	**10.1**	**23.9**
BASE+E2T$_{rnn+sd}$	27.6	7.9	21.5
BASE+E2T$_{cnn}$	26.6	7.3	20.7
BASE+E2T$_{rnn}$	26.1	6.9	20.1
BASE+E2T$_{cnn+soft}$	26.6	7.0	20.6
BASE+E2T$_{rnn+soft}$	25.0	6.7	19.8
Lead-3	26.1	9.6	17.8
LexRank	26.1	9.6	17.7
Bi-GRU	19.5	5.2	15.0
Distraction-M3	27.1	8.2	18.7
GBA	**30.3**	**9.8**	**20.0**

Table 3: Results on the CNN dataset using the full-length F1 ROUGE metric.

6 Results

We report the ROUGE F1 scores for both datasets of all the competing models using ROUGE F1 scores (Lin, 2004). We report the results on the Gigaword and the CNN dataset in Table 2 and Table 3, respectively. In Gigaword dataset where the texts are short, our best model achieves a comparable performance with the current state-of-the-art. In CNN dataset where the texts are longer, our best model outperforms all the previous models. We emphasize that E2T module is easily attachable

Model	1st	2nd	3rd	4th	mean
GOLD	0.27	0.34	0.21	0.18	2.38
BASE	0.14	0.15	0.28	0.43	3.00
BASE+E2T$_{rnn}$	0.12	0.24	0.39	0.25	2.77
BASE+E2T$_{cnn}$	**0.47**	0.27	0.12	0.14	**1.93**

Table 4: Human evaluations on the Gigaword dataset. Bold-faced values are the best while red-colored values are the worst among the values in the evaluation metric.

to better models, and we expect E2T to improve their performance as well. Overall, E2T achieves a significant improvement over the baseline model BASE, with at least 2 ROUGE-1 points increase in the Gigaword dataset and 6 ROUGE-1 points increase in the CNN dataset. In fact, all variants of E2T gain improvements over the baseline, implying that leveraging on linked entities improves the performance of the summarizer. Among the model variants, the CNN-based encoder with selective disambiguation and firm attention performs the best.

Automatic evaluation on the Gigaword dataset shows that the CNN and RNN variants of BASE+E2T have similar performance. To break the tie between both models, we also conduct human evaluation on the Gigaword dataset. We instruct two annotators to read the input sentence and rank the competing summaries from first to last according to their relevance and fluency: (a) the original summary GOLD, and from models (b) BASE, (c) BASE+E2T$_{cnn}$, and (d) BASE+E2T$_{rnn}$. We then compute (i) the proportion of every ranking of each model and (ii) the mean rank of each model. The results are reported in Table 4. The model with the best mean rank is BASE+E2T$_{cnn}$, followed by GOLD, then by BASE+E2T$_{rnn}$ and BASE, respectively. We also perform ANOVA and post-hoc Tukey tests to show that the CNN variant is significantly ($p < 0.01$) better than the RNN variant and the base model. The RNN variant does not perform as well as the CNN variant, contrary to the automatic ROUGE evaluation above. Interestingly, the CNN variant produces better (but with no significant difference) summaries than the gold summaries. We posit that this is due to the fact that the article title does not correspond to the summary of the first sentence.

Selective disambiguation of entities We show the effectiveness of the selective disambiguation gate d in selecting which entities to disambiguate or not. Table 6 shows a total of four different examples of two entities with the highest/lowest d values. In the first example, sentence **E1.1** contains the entity "*United States*" and is linked with the country entity of the same name, however the correct linked entity should be "*United States Davis Cup team*", and therefore is given a high d value. On the other hand, sentence **E1.2** is linked correctly to the country "*United States*", and thus is given a low d value.. The second example provides a similar scenario, where sentence **E2.1** is linked to the entity "*Gold*" but should be linked to the entity "*Gold medal*". Sentence **E2.2** is linked correctly to the chemical element. Hence, the former case received a high value d while the latter case received a low d value.

Entities as summary topic Finally, we provide one sample for each dataset in Table 5 for case study, comparing our final model that uses **firm** attention (BASE$_{cnn+sd}$), a variant that uses **soft** attention (BASE$_{cnn+soft}$), and the **baseline** model (BASE). We also show the attention weights of the **firm** and **soft** models.

In the Gigaword example, we find three observations. First, the base model generated a less informative summary, not mentioning "*mexico state*" and "*first edition*". Second, the soft model produced a factually wrong summary, saying that "*guadalajara*" is a mexican state, while actually it is a city. Third, the firm model is able to solve the problem by focusing only on the five most important entities, eliminating possible noise such as "*Unk*" and less crucial entities such as "*Country club*". We can also see the effectiveness of the selective disambiguation in this example, where the entity "*U.S. state*" is corrected to mean the entity "*Mexican state*" which becomes relevant and is therefore selected.

In the CNN example, we also find that the baseline model generated a very erroneous summary. We argue that this is because the length of the input text is long and the decoder is not guided as to which topics it should focus on. The soft model generated a much better summary, however it focuses on the wrong topics, specifically on "*Iran's nuclear program*", making the summary less general. A quick read of the original article tells us that the main topic of the article is all about the two political parties arguing over the deal with Iran. However, the entity "*nuclear*" appeared a lot in the article, which makes the soft model wrongly focus

Gigaword Dataset Example

Original	western mexico **@state** **@jalisco** will host the first edition of the **@UNK** dollar **@lorena ochoa** invitation **@golf** tournament on nov. ##-## #### , in **@guadalajara** **@country club** , the **@lorena ochoa** foundation said in a statement on wednesday .							
Gold	mexico to host lorena ochoa golf tournament in ####							
Baseline	guadalajara to host ochoa tournament tournament							
Entities:	U.S. state	Jalisco	Unk	Lorena Ochoa	Golf	Guadalajara	Country club	Lorena Ochoa
Soft	0.083	0.086	0.124	0.101	0.080	0.161	0.189	0.177
	mexico state guadalajara to host ochoa ochoa invitation							
Firm	0.173	0.197	0.000	0.213	0.215	0.000	0.00	0.202
	mexican state to host first edition of ochoa invitation							

CNN Dataset Example

Original	URL: http://edition.cnn.com/2015/04/05/politics/netanyahu-iran-deal/index.html
Gold	netanyahu says third option is " standing firm " to get a better deal . political sparring continues in u.s. over the deal with iran .
Baseline	netanyahu says he is a country of " UNK cheating " and that it is a country of " UNK cheating " netanyahu says he is a country of " UNK cheating " and that " is a very bad deal " he says he says he says the plan is a country of " UNK cheating " and that it is a country of " UNK cheating " he says the u.s. is a country of " UNK cheating " and that is a country of " UNK cheating "
Soft	benjamin netanyahu : " i think there 's a third alternative , and that is standing firm , " netanyahu tells cnn . he says he does not roll back iran 's nuclear ambitions . " it does not roll back iran 's nuclear program . "
Firm	new : netanyahu : " i think there 's a third alternative , and that is standing firm , " netanyahu says . obama 's comments come as democrats and republicans spar over the framework announced last week to lift western sanctions on iran .

Table 5: Examples from Gigaword and CNN datasets and corresponding summaries generated by competing models. The tagged part of text is marked **bold** and preceded with at sign (@). The red color fill represents the attention scores given to each entity. We only report the attention scores of entities in the Gigaword example for conciseness since there are 80 linked entities in the CNN example.

Text	d
Linked entity: https://en.wikipedia.org/wiki/United_States	
E1.1: andy roddick got the better of dmitry tursunov in straight sets on friday , assuring the **@united states** a #-# lead over defending champions russia in the #### davis cup final .	0.719
E1.2: sir alex ferguson revealed friday that david beckham 's move to the **@united states** had not surprised him because he knew the midfielder would not return to england if he could not come back to manchester united .	0.086
Linked entity: https://en.wikipedia.org/wiki/Gold	
E2.1: following is the medal standing at the ##th olympic winter games -lrb- tabulated under team , **@gold** , silver and bronze -rrb- : UNK	0.862
E2.2: **@gold** opened lower here on monday at ###.##-### .## us dollars an ounce , against friday 's closing rate of ###.##-### .## .	0.130

Table 6: Examples with highest/lowest disambiguation gate d values of two example entities (*United States* and *gold*). The tagged part of text is marked **bold** and preceded with at sign (@).

on the "*nuclear*" entity. The firm model produced the more relevant summary, focusing on the political entities (e.g. "*republicans*", "*democrats*"). This is due to the fact that only the $k = 10$ most important elements are attended to create the summary topic vector.

7 Conclusion

We proposed to leverage on linked entities to improve the performance of sequence-to-sequence models on neural abstractive summarization task. Linked entities are used to guide the decoding process based on the summary topic and commonsense learned from a knowledge base. We introduced Entity2Topic (E2T), a module that is easily attachable to any model using an encoder-decoder framework. E2T applies linked entities into the summarizer by encoding the entities with selective disambiguation and pooling them into one summary topic vector with firm attention mechanism. We showed that by applying E2T to a basic sequence-to-sequence model, we achieve significant improvements over the base model and consequently achieve a comparable performance with more complex summarization models.

Acknowledgement

We would like to thank the three anonymous reviewers for their valuable feedback. This work was supported by Microsoft Research, and Institute for Information communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2017-0-01778 , Development of Explainable Humanlevel Deep Machine Learning Inference Framework). S. Hwang is a corresponding author.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Diego Ceccarelli, Claudio Lucchese, Salvatore Orlando, Raffaele Perego, and Salvatore Trani. 2013. Dexter: an open source framework for entity linking. In *Proceedings of the sixth international workshop on Exploiting semantic annotations in information retrieval*. ACM, pages 17–20.

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, and Hui Jiang. 2016. Distraction-based neural networks for document summarization. *arXiv preprint arXiv:1610.08462* .

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252* .

Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. pages 93–98.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research* 22:457–479.

Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. 2016. On the reproducibility of the tagme entity linking system. In *European Conference on Information Retrieval*. Springer, pages 436–449.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*. pages 1693–1701.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580* .

Ioana Hulpus, Conor Hayes, Marcel Karnstedt, and Derek Greene. 2013. Unsupervised graph-based topic labelling using dbpedia. In *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM, pages 465–474.

Jey Han Lau, Paul Cook, and Timothy Baldwin. 2013. unimelb: Topic modelling-based word sense induction for web snippet clustering. In *SemEval@ NAACL-HLT*. pages 217–221.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*. Barcelona, Spain, volume 8.

Xiao Ling, Sameer Singh, and Daniel S Weld. 2015. Design challenges for entity linking. *Transactions of the Association for Computational Linguistics* 3:315–328.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* .

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023* .

David Newman, Chaitanya Chemudugunta, Padhraic Smyth, and Mark Steyvers. 2006. Analyzing entities and topics in news articles using statistical topic models. In *ISI*. Springer, pages 93–104.

Yuan Ni, Qiong Kai Xu, Feng Cao, Yosi Mass, Dafna Sheinwald, Hui Jia Zhu, and Shao Sheng Cao. 2016. Semantic documents relatedness using concept graph representation. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, pages 635–644.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. pages 1532–1543.

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685* .

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368* .

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research* 15(1):1929–1958.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. pages 3104–3112.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. 2017. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. volume 1, pages 1171–1181.

Jing Wang, Mohit Bansal, Kevin Gimpel, Brian D Ziebart, and T Yu Clement. 2015. A sense-topic model for word sense induction with unsupervised data enrichment. *Transactions of the Association for Computational Linguistics* 3:59–71.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*. pages 2048–2057.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2017. Learning distributed representations of texts and entities from knowledge base. *arXiv preprint arXiv:1705.02494* .

Jin-ge Yao, Xiaojun Wan, and Jianguo Xiao. 2017. Recent advances in document summarization. *Knowledge and Information Systems* pages 1–40.

Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. 2017. Selective encoding for abstractive sentence summarization. *arXiv preprint arXiv:1704.07073* .