

Unsupervised Sentence Compression using Denoising Auto-Encoders

Thibault Fevry*

Center for Data Science

New York University

Thibault.Fevry@nyu.edu

Jason Phang*

Center for Data Science

New York University

jasonphang@nyu.edu

Task



Figure 1: Illustration of Additive Noising. A reference sentence is noised with subsampled words from another sentence, and then shuffled. The denoising auto-encoder is trained to recover the original reference sentence. This simulates a text summarization training regimes without the need for parallel corpora.

Approach: Additive Noising

- Additive Sampling
 - Randomly sample two additional sentences
 - Subsample a number of words from each sentence
 - A noised sentence that extends the original sentence by 40% to 60%
- Shuffling
 - A complete word (unigram) shuffle
 - Bigram shuffling

Approach: Length Countdown

- Augment the RNN decoder to take an additional length countdown input
- $h_t = \text{RNN}(h_{t-1}, x_t) \rightarrow h_t = \text{RNN}(h_{t-1}, x_t, T - t)$
- $(x_t, T - t)$ are concatenated into a single vector

Approach: Input Sentence Embed

- Provide the model with an InferSent sentence embedding of the original sentence
- Initialise the hidden state of the decoder with the hidden state of the encoder's last position.
- from $\underline{h_0^{\text{dec}} = h_{T_{\text{enc}}}^{\text{enc}}}$ to $\underline{h_0^{\text{dec}} = f(h_{T_{\text{enc}}}^{\text{enc}}, s)}$

Approach: OOV Embeddings

- Given an input sequence, parse the sentence to identify OOV tokens and number them in order
- Also store the map from numbered OOV tokens to words
- During inference, replace any output numbered OOV tokens with their respective words.

Experiment

- Annotated Gigaword:
- Dataset statistic
 - 3.8M training examples and
 - 10K validation examples
 - 10K test examples

Main Results: ROUGE

Model	ROUGE			Avg. Length
	R-1	R-2	R-L	
<i>Baselines:</i>				
All text	28.91	10.22	25.08	31.3
F8W	26.90	9.65	25.19	8
<i>Unsupervised (Ours):</i>				
2-g shuf	27.72	7.55	23.43	15.4
2-g shuf + InferSent	28.42	7.82	24.95	15.6
<i>Supervised abstractive:</i>				
Seq2seq	35.50	15.54	32.45	15.4
(words-lvt2k-1sent) (Nallapati et al., 2016)	34.97	17.17	32.70	-

Table 1: Performance of Baseline, Unsupervised and Supervised Models. Our unsupervised models pale in comparison to supervised models, and perform in line with baselines. Simple baselines in text summarization benchmarks tend to be unusually strong. The unsupervised model incorporating sentence embeddings performs slightly better on ROUGE.

Ablation Study: ROUGE

Model	ROUGE		
	R-1	R-2	R-L
1-g shuf (w/o attn)	23.01	5.51	20.07
2-g shuf (w/o attn)	22.36	5.18	19.60
1-g shuf	27.22	7.63	23.55
2-g shuf	27.72	7.55	23.43
1-g shuf + InferSent	28.12	7.75	24.81
2-g shuf + InferSent	28.42	7.82	24.95

Table 2: Ablation study. We find that using attention, shuffling bigrams, and incorporating sentence embeddings all improve our ROUGE scores. All length countdowns settings are the same as in the main model.

Input Length	ROUGE			Avg. Length
	R-1	R-2	R-L	
16-30	30.79	9.20	27.73	12.6
31-45	26.89	6.76	23.04	17.7

Table 3: Effect of input sentence length on performance, using the 2-g shuf + InferSent model. Performance tends to be worse on longer input texts.

Main Results: Human

Model	Grammar	Meaning
2-g shuf	3.53 (± 0.18)	2.53 (± 0.16)
1-g shuf + InferSent	2.82 (± 0.17)	2.50 (± 0.15)
2-g shuf + InferSent	2.87 (± 0.16)	2.13 (± 0.13)
Seq2seq (Supervised)	3.43 (± 0.18)	2.60 (± 0.17)
Ground Truth	4.07 (± 0.13)	3.87 (± 0.16)

Table 4: Human Evaluation. Mean scores, with 1 standard error confidence bands in parentheses. Our best model performs competitively with a supervised baseline in both grammatical correctness and retention of meaning. Models with sentence embeddings perform worse in human evaluation, despite obtaining better ROUGE scores.

Unsupervised Statistical Machine Translation

Mikel Artetxe, Gorka Labaka, Eneko Agirre

IXA NLP Group

University of the Basque Country (UPV/EHU)

`{mikel.artetxe,gorka.labaka,e.agirre}@ehu.eus`

Background: Phrase-based SMT

- Phrase table
- Language model
- Reordering model
- Word and phrase penalties

Cross-lingual n-gram embeddings

- Skip gram for learning word embedding

$$\log \sigma (w \cdot c) + \sum_{i=1}^k \mathbb{E}_{c_N \sim P_D} [\log \sigma (-w \cdot c_N)]$$

- Skip gram for learning n-gram embedding
 - Learns the exact same embeddings as the original skip-gram for unigrams
 - Regards the n-grams as words.
- VecMap (Artetxe et al. (2018b)) for mapping two embeddings to a shared space

Unsupervised SMT

- Phrase table induction
- Unsupervised tuning
- Iterative refinement

USMT: Phrase Table Induction

- Phrase translation probabilities

$$\phi(\bar{f}|\bar{e}) = \frac{\cos(\bar{e}, \bar{f})/\tau}{\sum_{\bar{f}'} \cos(\bar{e}, \bar{f}')/\tau}$$

- Param learning

$$\min_{\tau} \sum_{\bar{f}} \log \phi(\bar{f} | \text{NN}_{\bar{e}}(\bar{f})) + \sum_{\bar{e}} \log \phi(\bar{e} | \text{NN}_{\bar{f}}(\bar{e}))$$

USMT: Unsupervised Tuning

Algorithm 1 Unsupervised tuning

Input: $m_{s \rightarrow t}$ (source-to-target models)

Input: $m_{t \rightarrow s}$ (target-to-source models)

Input: c_s (source validation corpus)

Input: c_t (target validation corpus)

Output: $w_{s \rightarrow t}$ (source-to-target weights)

Output: $w_{t \rightarrow s}$ (target-to-source weights)

1: $w_{t \rightarrow s} \leftarrow \text{DEFAULT_WEIGHTS}$

2: **repeat**

3: $bt_s \leftarrow \text{TRANSLATE}(m_{t \rightarrow s}, w_{t \rightarrow s}, c_t)$

4: $w_{s \rightarrow t} \leftarrow \text{MERT}(m_{s \rightarrow t}, bt_s, c_t)$

5: $bt_t \leftarrow \text{TRANSLATE}(m_{s \rightarrow t}, w_{s \rightarrow t}, c_s)$

6: $w_{t \rightarrow s} \leftarrow \text{MERT}(m_{t \rightarrow s}, bt_t, c_s)$

7: **until** convergence

USMT: Iterative refinement

- Limitations in Unsupervised Tuning
 - It does not use any lexical reordering model
 - Its phrase table is limited by the embedding vocabulary
 - The phrase translation probabilities and lexical weightings are estimated based on cross-lingual embeddings

USMT: Iterative refinement

Algorithm 1 Unsupervised tuning

Input: $m_{s \rightarrow t}$ (source-to-target models)

Input: $m_{t \rightarrow s}$ (target-to-source models)

Input: c_s (source validation corpus)

Input: c_t (target validation corpus)

Output: $w_{s \rightarrow t}$ (source-to-target weights)

Output: $w_{t \rightarrow s}$ (target-to-source weights)

- 1: $w_{t \rightarrow s} \leftarrow \text{DEFAULT_WEIGHTS}$
 - 2: **repeat**
 - 3: $bt_s \leftarrow \text{TRANSLATE}(m_{t \rightarrow s}, w_{t \rightarrow s}, c_t)$
 - 4: $w_{s \rightarrow t} \leftarrow \text{MERT}(m_{s \rightarrow t}, bt_s, c_t)$
 - 5: $bt_t \leftarrow \text{TRANSLATE}(m_{s \rightarrow t}, w_{s \rightarrow t}, c_s)$
 - 6: $w_{t \rightarrow s} \leftarrow \text{MERT}(m_{t \rightarrow s}, bt_t, c_s)$
 - 7: **until** convergence
-

Algorithm 2 Iterative refinement

Input: c_s (source language corpus)

Input: c_t (target language corpus)

Input/Output: $m_{t \rightarrow s}$ (target-to-source models)

Input/Output: $w_{t \rightarrow s}$ (target-to-source weights)

Output: $m_{s \rightarrow t}$ (source-to-target models)

Output: $w_{s \rightarrow t}$ (source-to-target weights)

- 1: $train_s, val_s \leftarrow \text{SPLIT}(c_s)$
 - 2: $train_t, val_t \leftarrow \text{SPLIT}(c_t)$
 - 3: **repeat**
 - 4: $btt_s \leftarrow \text{TRANSLATE}(m_{t \rightarrow s}, w_{t \rightarrow s}, train_t)$
 - 5: $btt_s \leftarrow \text{TRANSLATE}(m_{t \rightarrow s}, w_{t \rightarrow s}, val_t)$
 - 6: $m_{s \rightarrow t} \leftarrow \text{TRAIN}(btt_s, train_t)$
 - 7: $w_{s \rightarrow t} \leftarrow \text{MERT}(m_{s \rightarrow t}, btt_s, val_t)$
 - 8: $btt_t \leftarrow \text{TRANSLATE}(m_{s \rightarrow t}, w_{s \rightarrow t}, train_s)$
 - 9: $btt_t \leftarrow \text{TRANSLATE}(m_{s \rightarrow t}, w_{s \rightarrow t}, val_s)$
 - 10: $m_{t \rightarrow s} \leftarrow \text{TRAIN}(btt_t, train_s)$
 - 11: $w_{t \rightarrow s} \leftarrow \text{MERT}(m_{t \rightarrow s}, btt_t, val_s)$
 - 12: **until** convergence
-

Experiments

	WMT-14				WMT-16	
	FR-EN	EN-FR	DE-EN	EN-DE	DE-EN	EN-DE
Artetxe et al. (2018c)	15.56	15.13	10.21	6.55	-	-
Lample et al. (2018)	14.31	15.05	-	-	13.33	9.64
Yang et al. (2018)	15.58	16.97	-	-	14.62	10.86
Proposed system	25.87	26.22	17.43	14.08	23.05	18.23

Table 1: Results of the proposed method in comparison to existing unsupervised NMT systems (BLEU).

	WMT-14				WMT-16	
	FR-EN	EN-FR	DE-EN	EN-DE	DE-EN	EN-DE
Unsupervised SMT	21.16	20.13	13.86	10.59	18.01	13.22
+ unsupervised tuning	22.17	22.22	14.73	10.64	18.21	13.12
+ iterative refinement (it1)	24.81	26.53	16.01	13.45	20.76	16.94
+ iterative refinement (it2)	26.13	26.57	17.30	13.95	22.80	18.18
+ iterative refinement (it3)	25.87	26.22	17.43	14.08	23.05	18.23

Table 2: Ablation results (BLEU). The last row corresponds to our full system. Refer to the text for more details.

Experiments

		WMT-14				WMT-16	
		FR-EN	EN-FR	DE-EN	EN-DE	DE-EN	EN-DE
Supervised	NMT (transformer)	-	41.8	-	28.4	-	-
	WMT best	35.0	35.8	29.0	20.6	40.2	34.2
	SMT (europarl)	30.61	30.82	20.83	16.60	26.38	22.12
	+ w/o lexical reord.	30.54	30.33	20.37	16.34	25.99	22.20
	+ constrained vocab.	30.04	30.10	19.91	16.32	25.66	21.53
	+ unsup. tuning	29.32	29.46	17.75	15.45	23.35	19.86
Unsup.	Proposed system	25.87	26.22	17.43	14.08	23.05	18.23

Table 3: Results of the proposed method in comparison to supervised systems (BLEU). Transformer results reported by [Vaswani et al. \(2017\)](#). SMT variants are incremental (e.g. 2nd includes 1st). Refer to the text for more details.