Noisy Self-Knowledge Distillation for Text Summarization

Yang Liu[1] , Sheng Shen[2] and Mirella Lapata[3]

[1]Microsoft Cognitive Services Research
[2]University of California, Berkeley
[3]School of Informatics, University of Edinburgh

Challenges in Modern Summarization Systems

Human variation in summarization tasks

- In the context of extractive summarization, different people select different sentences to include in a summary [1].
- When writing abstracts, disagreement exists both in terms of writing style and the specific content deemed important for the summary [2].

Most popular benchmarks are collated opportunistically

- In many popular summarization datasets, summaries are only loosely correspond to the source input.
- [3] pair articles with metadata available in HTML pages under the assumption that HTML tags denote summary-like content.
- [4] use lead sections in Wikipedia articles as summaries of documents cited therein.

The inherent *noise* in the data collection process further hampers training with models often being prone to hallucination [5], and struggling to identify which content units are salient.

In this work, we propose to alleviate these problems by turning to **knowledge distillation**, where outputs provide softened distributions of the reference summaries as:

- An enrichment of the single reference setting
- A reweighting of gold summaries

Knowledge Distillation

Knowledge Distillation refers to a class of methods for training a new smaller *student* network by learning from a *teacher* network. Let T and S denote teacher and student models, respectively. Also, let f_T and f_S be functions of the teacher and student. The models are typically neural networks and function f can be in principle defined using the output of any network layer (e.g. softmax layer). Knowledge distillation methods are commonly expressed as minimizing an objective function over training set $\mathcal{X}$:

$$\mathcal{L}_{KD} = \sum_{x_i \in \mathcal{X}} l(f_T(x_i), f_S(x_i)) \tag{1}$$

where $l()$ is a loss function that penalizes the difference between the teacher and the student.

Self-Knowledge Distillation for Text Summarization

Self-knowledge distillation refers to the special case where the teacher and student have *identical* neural network architectures. The standard objective for an abstractive summarization model is negative log likelihood:

$$\mathcal{L}_{\text{NLL}} = -\sum_{t=1}^{T} log(p(y_t|y_1^{t-1}, x)) \tag{2}$$

where x indicates the source document, y_1^t indicates the t-th token in the target summary and y_1^{t-1} are the first $t-1$ tokens in the target summary.

We further assume that the teacher is a fully trained neural model, the student has the same architecture with the teacher and access to the learned teacher's output distribution, the distillation loss is:

$$\mathcal{L}_{\text{KD}} = \sum_{t=1}^{T} \text{KL}(p_T(y_t|y_1^{t-1}, x), p_S(y_t|y_1^{t-1}, x)) \tag{3}$$

where $p_T(y_t|y_1^{t-1}, x)$ and $p_S(y_t|y_1^{t-1}, x)$ are model outputs from the teacher and student, respectively.

It is common practice to compensate for no direct access to the training data by interpolating between the two losses in Equations (3) and (2). So, the final objective for training the student becomes:

$$\mathcal{L}_{\text{FINAL}} = (1-\lambda)\mathcal{L}_{\text{NLL}} + \lambda\mathcal{L}_{\text{KD}} \tag{4}$$

where λ is a mixture parameter combining one-hot distribution and teacher distribution.

Noise Injection for Self-Knowledge Distillation

We further want our summarization systems to be robust to natural noise found in existing datasets. Injecting noise onto training process has been proven useful for improving model generalization.

Noisy Teacher

To inject noise into the distillation signals, we incorporate a **teacher dropout** mechanism, where dropout is kept active while generating teacher predictions for training the student. This has two advantages:

- The teacher generates variable supervision labels
- The teacher can also be considered as approximating an average ensemble from many neural networks

With dropout rate α, the knowledge distillation loss now becomes:

$$\mathcal{L}_{\text{KD}} = \sum_{t=1}^{T} \text{KL}(\tilde{p}_T^{\alpha}(y_t|y_1^{t-1}, x), p_S(y_t|y_1^{t-1}, x)) \tag{5}$$

where $\tilde{p}_T^{\alpha}$ indicates the predictions from the teacher model with active dropout.

Noisy Student

To inject noise into the training data, we propose various mechanisms to perturb the source input:

1. *Word Drop*: a word in the source input is removed with probability p_d.
2. *Word Replacement*: for each word x_i in the source input, we calculate a candidate replacement list by selecting k words most similar to x_i.
3. *Sentence Drop*: a sentence in the source input is removed with probability p_s.
4. *Gaussian Noise*: a Gaussian noise vector $\mathbf{e}$ is multiplied with the embeddings $\mathbf{x}$ of input words: $\mathbf{x} \leftarrow \mathbf{x} \otimes \mathbf{e}, \mathbf{e} \sim N(I, \sigma^2 I)$.

The knowledge distillation loss with a student trained on noisy data becomes:

$$\mathcal{L}_{\text{KD}} = \sum_{t=1}^{T} \text{KL}(\tilde{p}_T^{\alpha}(y_t|y_1^{t-1}, x), p_S(y_t|y_1^{t-1}, \tilde{x})) \tag{6}$$

where $\tilde{x}$ indicates perturbed source input.

Experiments

Experiments are done on single-document datasets CNN/DM and XSum and multi-document dataset WikiCatSum.

Without Pretraining		CNN/DailyMail R1	CNN/DailyMail R2	CNN/DailyMail RL	XSum R1	XSum R2	XSum RL
LEAD		40.42	17.62	36.67	16.30	1.60	11.95
PtrNet		39.53	17.28	36.38	28.10	8.02	21.72
TransformerAbs		40.21	17.76	37.09	31.04	10.48	24.54
+SKD		40.64	18.10	37.43	32.22	11.45	25.56
+SKD +Noisy T		40.79	18.24	37.57	32.32	11.56	25.72
+SKD +Noisy T +Noisy S		40.86	18.27	37.66	32.76	11.88	26.07
BASE-size Pretrained Models		R1	R2	RL	R1	R2	RL
$\text{MASS}_{\text{BASE}}$	(123M)	42.12	19.50	39.01	39.75	17.24	31.95
BERTSumAbs	(156M)	41.72	19.39	38.76	38.76	16.33	31.15
$\text{UniLMv2}_{\text{BASE}}$	(110M)	43.45	20.71	40.49	43.69	20.71	35.73
+SKD	(110M)	43.44	20.68	40.51	43.76	21.04	36.04
+SKD +Noisy T	(110M)	43.59	21.01	40.66	44.11	21.30	36.32
+SKD +Noisy T +Noisy S	(110M)	43.77	20.98	40.82	44.14	21.34	36.35
LARGE-size Pretrained Models		R1	R2	RL	R1	R2	RL
$\text{UniLM}_{\text{LARGE}}$	(340M)	43.08	20.43	40.34	---	---	---
$\text{BART}_{\text{LARGE}}$	(400M)	44.16	21.28	40.90	45.14	22.27	37.25
T5_{11B}	(11B)	42.05	20.34	39.40	---	---	---

Table 1: ROUGE F1 results on **CNN/DM** and **XSum** test sets

Without Pretraining	R1	R2	RL
CV-S2S	33.8	19.2	29.7
CV-S2D	35.9	19.5	30.1
TF-S2S	35.5	19.0	30.5
+SKD	36.1	19.4	31.0
+SKD +Noisy T	36.5	20.0	31.1
+SKD +Noisy T +Noisy S	36.6	20.1	31.3
With Pretraining	R1	R2	RL
$\text{UniLMv2}_{\text{BASE}}$	40.4	24.0	34.3
+SKD	40.4	24.1	34.4
+SKD +Noisy T	40.6	24.4	34.6
+SKD +Noisy T +Noisy S	40.7	24.4	34.7

Table 2: ROUGE F1 results on **WikiCatSum** test sets

References

[1] GJ Rath, A Resnick, and TR Savage.
The formation of abstracts by the selection of sentences. part i. sentence selection by men and machines.

[2] Donna Harman and Paul Over.
The effects of human variation in DUC summarization evaluation.

[3] Max Grusky, Mor Naaman, and Yoav Artzi.
Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies.

[4] Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer.
Generating Wikipedia by summarizing long sequences.

[5] Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald.
On faithfulness and factuality in abstractive summarization.