Frustratingly Easy Model Ensemble for Abstractive Summarization

Hayato Kobayashi
Yahoo Japan Corporation / RIKEN AIP

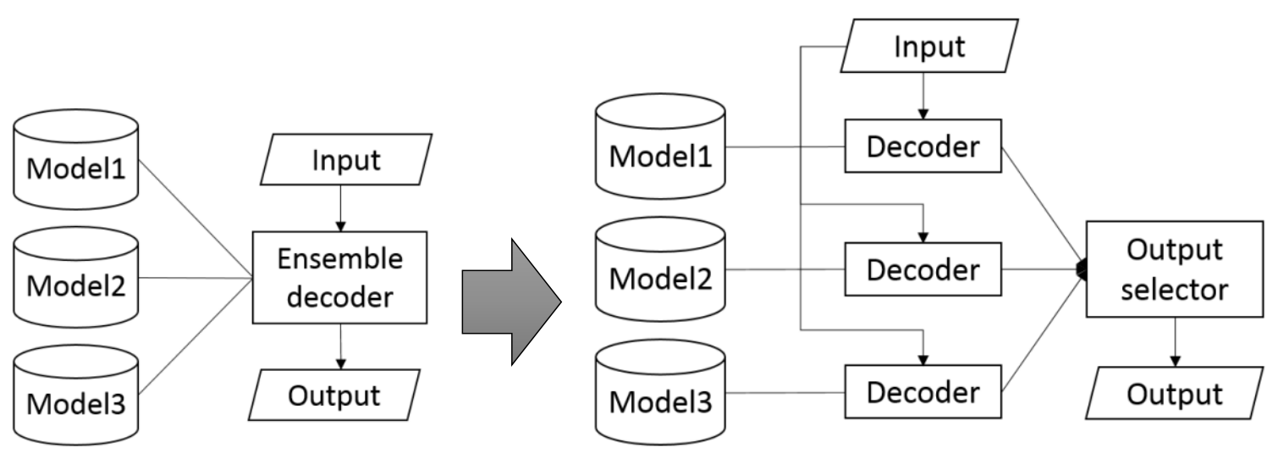
Introduction

Background

- Model ensemble is known to be effective for text generation
  - Drawback: More computational costs ("# of models" times)
- Many studies on compression of an ensemble model (Hinton+ 2015, …)
  - Drawback: Cannot be easily applied to other models

Approach

- Selects a majority-like output from generated outputs in post-processing, inspired by the majority vote in classification tasks



(a) Runtime-ensemble (b) Post-ensemble

Contributions

- Propose a new model ensemble method (post-ensemble)
  - Simple: Implementation is "frustratingly easy" w/o model code change
  - Fast: Computational time is enough for practical use (3.7 ms/sent)
  - Effective: Performance is better than runtime-ensemble
- Prove a relationship to kernel density estimation based on vMF kernel

Preliminaries

Encoder-Decoder Model

- Conditional language model that learns an appropriate output sequence y given a sequence x from a lot of pairs (x, y)

$$p(y \mid x) = \prod_{t=1}^{T-1} p(\overbrace{y_{t+1}}^{(t+1)\text{-th word in } y} \mid \underbrace{y_{\leq t}}_{\text{Seq. from } y_1 \text{ to } y_{t-1}}, x)$$

Runtime-Ensemble

- Averages word prediction probabilities p in each decoding step

Arithmetic mean (EnsSum)

$$\overline{p}_{\mathrm{a}}(y_{\leq t} \mid x) = \frac{1}{|M|} \sum_{p \in M} p(y_{\leq t} \mid x),$$

Geometric mean (EnsMul)

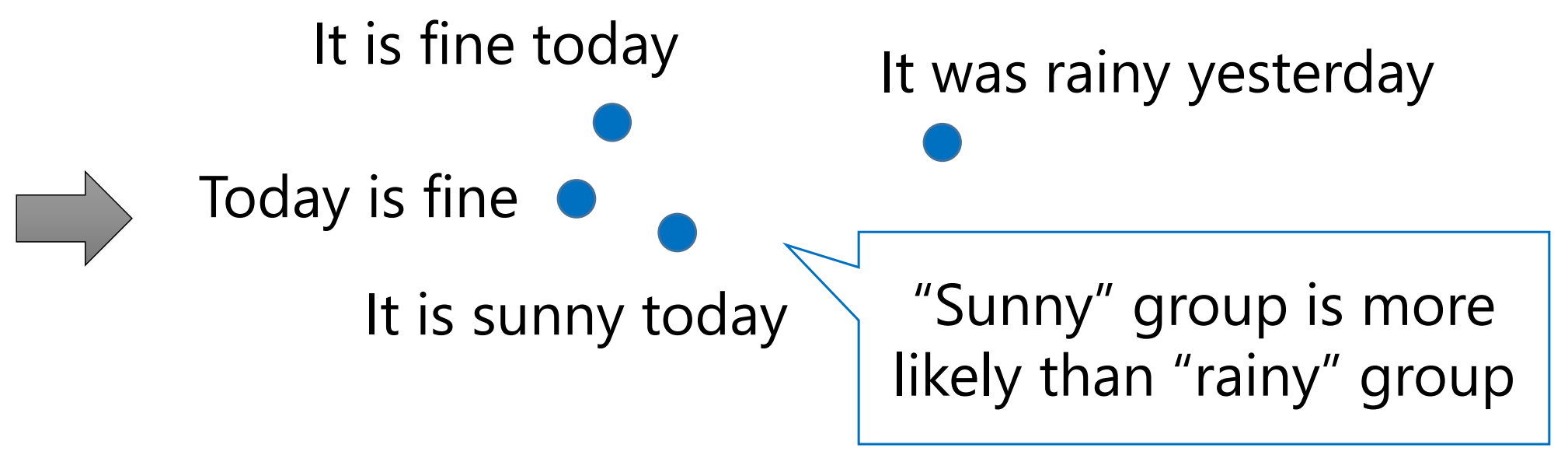
$$\overline{p}_{\mathrm{g}}(y_{\leq t} \mid x) = \left(\prod_{p \in M} p(y_{\leq t} \mid x) \right)^{\frac{1}{|M|}}$$

Post-Ensemble

Difficulty and Idea

- Difficulty: Majority output may not exist for text-generation
  - Text is a sequence of labels, not a label
- Idea: Instead of exact match frequency, use cosine similarity

Output	Freq.
It is fine today	1
It is sunny today	1
It was rainy yesterday	1
Today is fine	1



"Sunny" group is more likely than "rainy" group

Algorithm

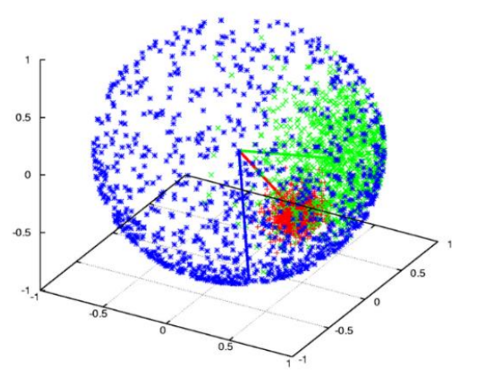
- Select a majority-like output based on cosine sim. in post-processing

```
Input: Input text $x$, set $M$ of learned models, and
       similarity function $K$, such as cos.
Output: Output prediction $y$.
$S \leftarrow \emptyset$;
foreach $p \in M$ do
    $s \leftarrow$ output of model $p$ for input $x$;
    $S \leftarrow S \cup \{s\}$;
$C \leftarrow \{\}$; // as a hash map
foreach $s \in S$ do
    $c \leftarrow \frac{1}{|S|} \sum_{s' \in S} K(s, s')$;
    $C[s] \leftarrow c$;
$y = \mathrm{argmax}_{s \in S}\, C[s]$;
return $y$
```

Algorithm 1: Post-ensemble procedure.

- Generate an output on each decoder
  - No code modifications
  - Easily parallelizable

- Select the closest one to the other outputs by cosine similarity



Theoretical Analysis

- Post-ensemble is an approx. of KDE based on vMF kernel

Theorem 1. The output y of Alg. 1 with $K(s, s') = \cos(s, s')$ is equivalent to the maximization of the first order Taylor series approximation $\tilde{p}$ of the kernel density estimator p based on the von Mises-Fisher kernel, i.e.,

$$\tilde{p}(y) = \max\nolimits_{s \in S} \tilde{p}(s), \tag{8}$$

where the approximation error R^* of the output y with respect to the true density estimator p, i.e,. $R^* = \max_{s \in S} p(s) - p(y)$, is bounded by

$$R^* \leq C_q(\kappa) \kappa^2 \exp(\kappa)(\sigma^2 + \mu^2), \tag{9}$$

where $\mu = \max_{s \in S} \mathbb{E}_{s'}[\cos(s, s')]$, and $\sigma^2 = \max_{s \in S} \mathbb{V}_{s'}[\cos(s, s')]$.

Kernel Density Estimation (KDE)

- Non-parametric method to estimate a probability density $\tilde{f}(X) = \frac{1}{n} \sum_{i=1}^{n} K(X, X_i)$.

von Mises-Fisher (vMF) Kernel

- Natural variant of Gaussian kernel to a unit hypersphere
- Compatible with cosine similarity

$$K_{\mathrm{vmf}}(s, s') = C_q(\kappa) \exp(\kappa \cos(s, s')),$$

$$C_q(\kappa) = \left(\kappa^{\frac{q-1}{2}}\right) \Big/ \left((2\pi)^{\frac{q+1}{2}} \mathcal{I}_{\frac{q-1}{2}}(\kappa)\right), \quad \kappa = \hat{h}^{-2}$$

$$\hat{h} = \left(\frac{4\pi^{\frac{1}{2}} \mathcal{I}_{\frac{q-1}{2}}(\tilde{\kappa})^2}{\tilde{\kappa}^{\frac{q+1}{2}} \left(2q \mathcal{I}_{\frac{q+1}{2}}(2\tilde{\kappa}) + (2+q) \tilde{\kappa} \mathcal{I}_{\frac{q+3}{2}}(2\tilde{\kappa}) \right) n} \right)^{\frac{1}{4+q}}$$

Experiments

Basic Settings

- Dataset: Gigaword's news headline generation task (Rush+, 2015)
  - Consists of (first sentence, headline) pairs (3.8M/189K/2K)
- Model: Encoder-decoder model with global attention
  - Default implementation of OpenNMT-py (with 500-dim units)
  - Prepared 10 models by random initialization
- Decoding: Beam search with width 5 (UNK-replaced)
- Evaluation: ROUGE-1, -2, -L (Averaged over 10 experiments)

Compared Models

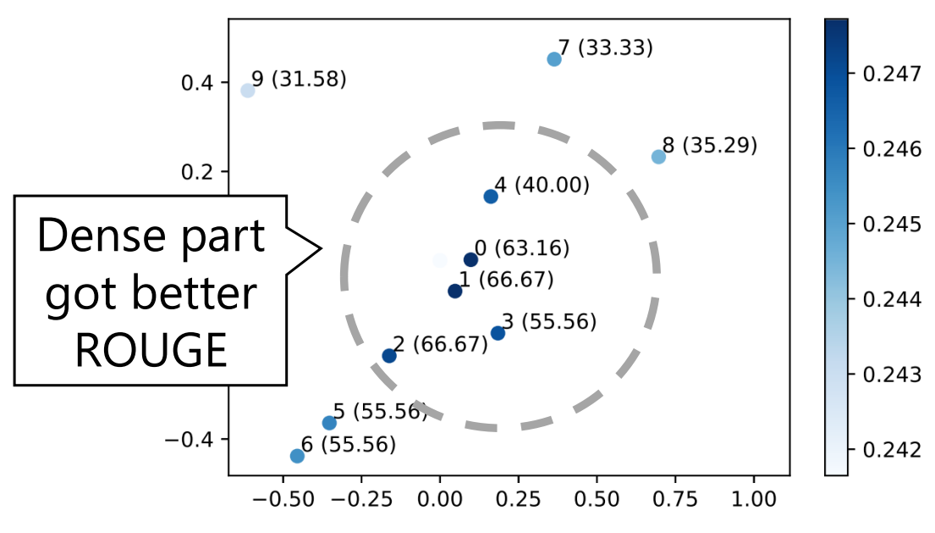
- Single
  - Best single model w.r.t. word level accuracy on a validation set
- EnsSum, EnsMul
  - Runtime-ensemble by averaging with summation / multiplication
- MaxLik, MajVote
  - Selects each output by its likelihood / exact majority vote
- LexRank, LMRank
  - LexRank (Erkan&Radev, 2004) / LM reranking (Vaswani+, 2013)
- PostCos(E/B), PostVmf(E/B)
  - Post-ensemble with cosine or vMF (Emb. / BoW features)

Main Results

- PostCos performed better than baselines

	R-1	R-2	R-L
Single	35.57	17.47	33.19
EnsSum	36.55	18.48	34.24
EnsMul	36.47	18.35	34.16
MaxLik	35.04	17.37	32.80
MajVote	35.97	18.09	33.67
LexRank	36.03	17.64	33.60
LMRank	35.07	17.12	32.61
PostCosE	37.02	18.46	34.54
PostVmfE	**37.06**	18.53	34.60
PostCosB	37.05	**18.59**	**34.61**
PostVmfB	37.02	18.58	34.59
MaxRef*	45.40	24.61	42.09
Mean*	35.57	17.48	33.19
Max*	36.03	17.83	33.63
Min*	35.00	17.08	32.67

How does post-ensemble work?



Dense part got better ROUGE

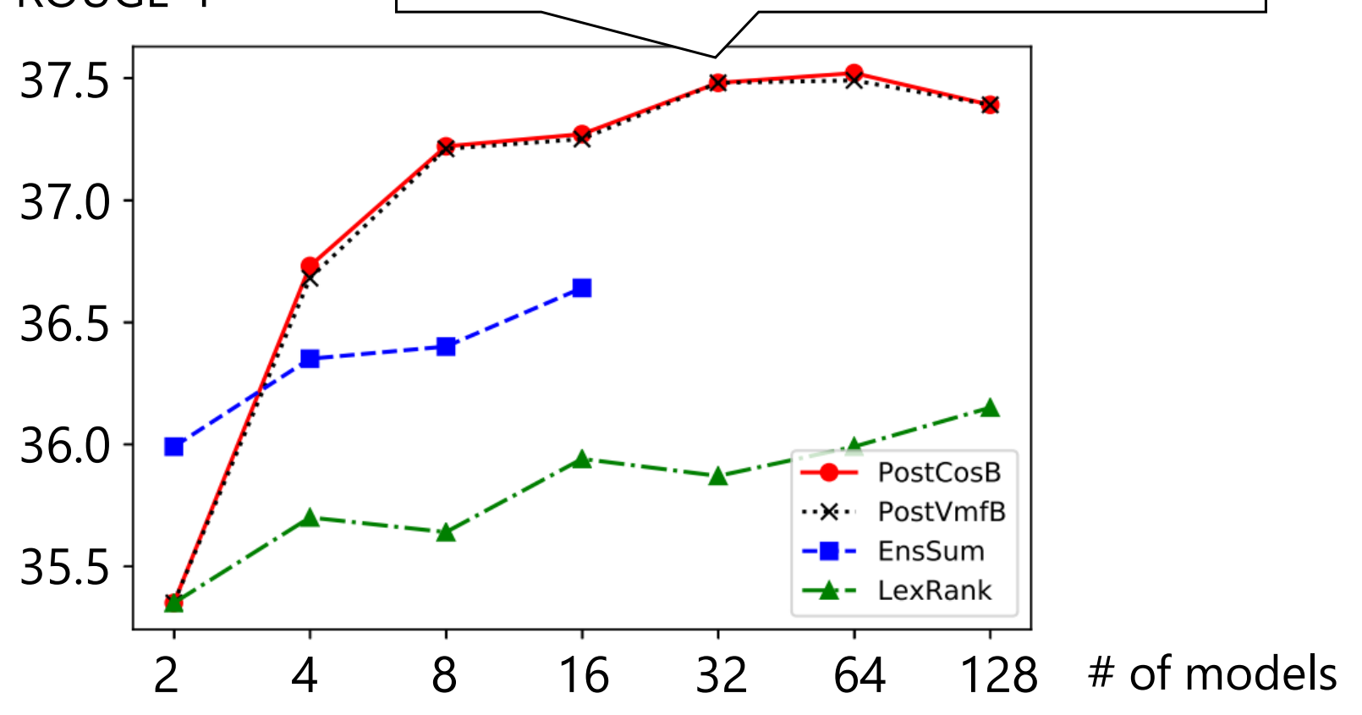
0: **interpol asks** members **to** devise **rules for policing** …
1: **interpol asks** members **to** devise **rules for policing**
2: **interpol asks** members **to** devise **rules for policing** at …
3: **interpol asks** members **to** devise **rules** on **policing**
4: **interpol asks** members **to** devise **rules** and procedures …
5: **interpol** seeks **rules for policing** of **global** level
6: **interpol** seeks **rules for policing** at **global** level
7: **interpol asks** members **to** act against wanted fugitives
8: **interpol asks** members **to** help fight fugitives
9: **interpol asks for** legal status **for** red corner notices

Figure 2: Left scatter-plot shows two-dimensional visualization of outputs generated from 10 models on basis of multi-dimensional scaling (Cox and Cox, 2008), and right list shows their contents. Each point in plot represents sentence embedding of corresponding output, and label indicates model ID and ROUGE-1, i.e., "ID (ROUGE)." Color intensity means score of kernel density estimation of PostCosE (see right color bar), and outputs are sorted by scores. Reference and input are as follows. Each bold word in above list means co-occurrence with reference below.
Reference: *interpol asks world govts to make rules for global policing*
Input: *top interpol officers on wednesday asked its members to devise rules and procedures for policing at the global level and providing legal status to red corner notices against wanted fugitives .*

Effect of Number of Models

- May be saturated after 32 models
- Computational time is reasonable
  - Ensemble time: 3.7 ms/sentence
  - Decoding time: 44 ms/sentence
- Runtime-ensemble couldn't be calculated with over 16 models
- PostCos and PostVmf have almost the same tendency (cf. Theorem 1)

	R-1	R-2	R-L
PostCosE (32 models)	37.52	18.55	34.86
PostCosB (32 models)	37.48	18.76	34.99
(Zhou et al., 2017)♯	36.15	17.54	33.63
(Suzuki and Nagata, 2017)♯	36.30	17.31	33.88
(Cao et al., 2018)♯	37.27	17.65	34.24



Effect of Model Preparation

- Random: Randomly initialize parameters
- Self: Use the last 10 epochs in a single run
- Hetero: Use 10 model structures
  - 8 models by replacing default settings with {2,3} layers, {250,500} LSTM dim., {250,500} embedding dim.
  - 2 models by replacing BiLSTM with an LSTM and BiLSTM with different merge
- Bagging: Extract 80% from a training set

	Random	Self	Hetero	Bagging
Single	35.57	35.34	35.67	34.87
EnsSum	36.55	35.46	36.42	36.25
EnsMul	36.47	35.22	36.49	35.80
MaxLik	35.04	34.21	34.86	34.95
MajVote	35.97	35.49	35.89	35.22
LexRank	36.03	33.57	35.91	35.72
LMRank	35.07	33.47	34.71	34.39
PostCosE	37.02	**35.91**	36.57	**36.89**
PostVmfE	**37.06**	35.72	36.69	36.84
PostCosB	37.05	35.74	**36.76**	36.78
PostVmfB	37.02	35.75	36.75	36.81
MaxRef*	45.40	43.37	45.32	46.44
Mean*	35.57	34.43	35.28	34.85
Max*	36.03	35.34	35.96	35.31
Min*	35.00	33.43	34.49	34.36

Discussion

Relationships to Existing Communities

- New category of algorithms for model ensemble
  - Main: Model selection in preprocessing, model average at run-time
  - New: Output selection in post-processing
- New category of tasks for hypothesis reranking
  - Main: Select one from the N-best hypotheses of a single model
  - New: Select one from the best outputs of multiple models

Future Work

- Learning of kernel functions (metric learning)
- Learning to rank outputs (list-wise reranking)
- Active learning to select a new model structure
- Boosting-like-ensemble extending bagging-ensemble