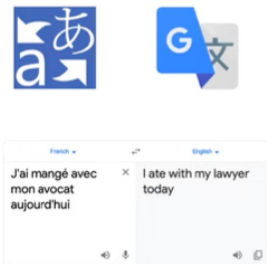


Lecture12 - Natural Language Generation

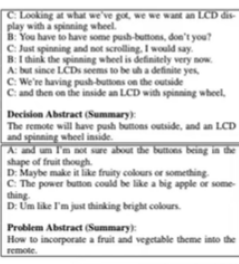
▼ What is NLG

- NLG → Natural Language Generation (NLP 분야 중 하나)
 - Machine Translation, Dialogue System, Summarization, Data-to-Text Generation, Visual Description
 - 입력 데이터에 따라 적절한 단어 생성하는 모든 task 의미

<Machine Translation>

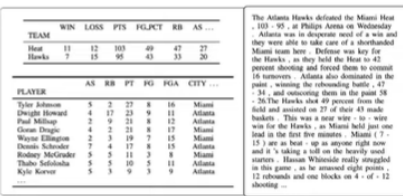


<Summarization>



{Wang and Cardie, ACL 2013}

<Data-to-Text Generation>



{Wiseman and Rush, EMNLP 2017}

- 사람의 편의를 위해 text 생성하는 모든 업무 → NLG 사용 가능
- Formalizing
 - Basic NLG
 - Autoregressive 형태의 일반적인 NLG 모델 → 이전까지의 단어 $\{y\}_{<t}$ 를 입력 받아서 다음 스텝의 단어 \hat{y}_t 를 생성
 - 학습시엔 예측된 토큰과 실제 토큰을 사용한 Negative loglikelihood를 minimize 하는 방식으로 사용
 - teacher forcing

The diagram illustrates a Text Generation Model during the Teacher Forcing phase. A central grey rounded rectangle is labeled "Text Generation Model". Above the model, a sequence of blue labels represents the target outputs: y_1^* , y_2^* , y_3^* , y_4^* , followed by an ellipsis, then y_{T-3}^* , y_{T-2}^* , y_{T-1}^* , and y_T^* . A blue label "<END>" is positioned above the final output y_T^* . Below the model, a sequence of red labels represents the inputs: y_0^* , y_1^* , y_2^* , y_3^* , followed by an ellipsis, then y_{T-4}^* , y_{T-3}^* , y_{T-2}^* , and y_{T-1}^* . Vertical arrows point from each input label to the model, and from the model to each output label.

- ## ▼ Decoding

- ### ▼ decoding 방식 2 종류 (Argmax Decoding, Beam Search)

Argmax Decoding

The diagram illustrates the Argmax Decoding process. It shows a sequence of hidden states (green boxes) connected by arrows. Each state has an 'argmax' label above it. Below each state is a set of input tokens. Pink dashed arrows connect the 'argmax' output of one state to the input of the next state. The sequence of tokens is: **<START>**, *he*, *hit*, *me*, *with*, *a*, *pie*, **<END>**. The sequence of outputs is: *he*, *hit*, *me*, *with*, *a*, *pie*.

Beam Search Tree Diagram:

- <START> (-0.7)
 - he (-0.7)
 - hit (-1.7)
 - a (-2.8)
 - tart (-4.0)
 - in (-4.8)
 - pie (-4.3)
 - tart (-4.6)

For each of the k hypotheses, find top k next words and calculate scores

- Beam Search

- 2

- 매 step 마다 vocab 내의 단어 중 probability가 최대가 되는 단어를 선택

▼ Greedy Methods 문제점

- 비슷한 문장들 반복
 - 작은 step 단위 별로 최적의 선택을 했던 것이 좀 더 큰 step에서 보면 부자연스러운 문장 생성하게 된다.

Context: In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

Continuation: The study, published in the Proceedings of the National Academy of Sciences of the United States of America (PNAS), was conducted by researchers from the **Universidad Nacional Autónoma de México (UNAM)** and **the Universidad Nacional Autónoma de México (UNAM/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México...**

(Holtzman et. al., ICLR 2020)

- Greedy decoding 방식은 동일한 phrase를 반복해서 생성하는 문제 존재
 - Dialogue task 나 chat bot system 같은 open end 문장을 생성할 때 주로 발생
- GPT 에선 특정 표현이 반복 생성될 때, 생성 결과에 대해 모델의 confidence 상승 (Bottleneck을 없앤 attention 구조로 인해)



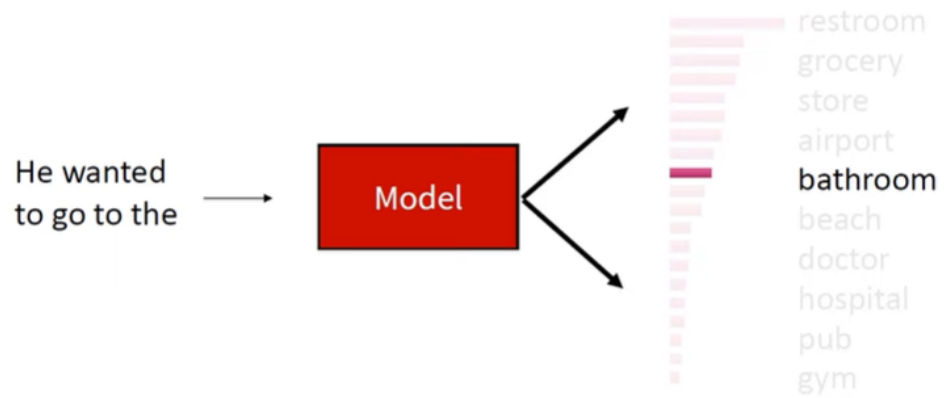
어떻게 해결할 수 있을까

- Decoding 단계에서 n-grams이 반복을 막음

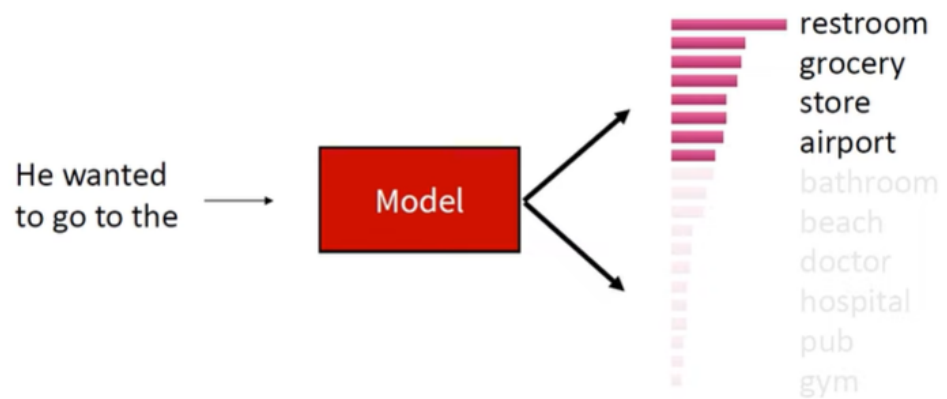
▼ Decoding 알고리즘 변경

1. Random sampling

- a. 모델이 예측한 token distribution을 sampling에 사용 → 어떤 단어든 (prob≠0) 등장할 수 있음 !!

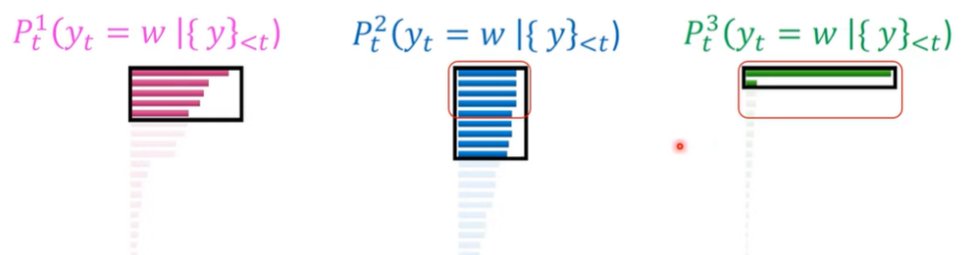


2. Top-k sampling



1. random sampling 시 낮은 확률로 선택된 단어로 인해 부자연스러운 문장 생성 가능 → Top -k sampling 사용
2. 모델이 예측한 token distribution에서 상위 k개에서만 sampling (ex k = 5, 10, 15) → hyperparameter
3. k 클수록 : diverse/risky outputs(부자연스러운)
4. k 작을수록 : generic/safe outputs(자연스러운)

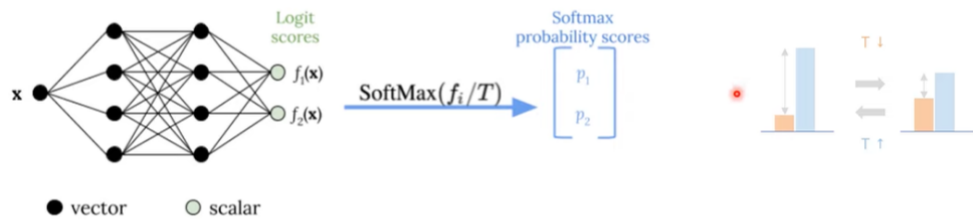
3. Top-p sampling



- 누적 확률 값이 p 보다 작은 상위 토큰들만 샘플링에 사용 $p \rightarrow$ hyperparameter
- p 클수록 : diverse/risky outputs
- p 작을수록 : generic/safe outputs

4. + Scaling randomness : Temperature

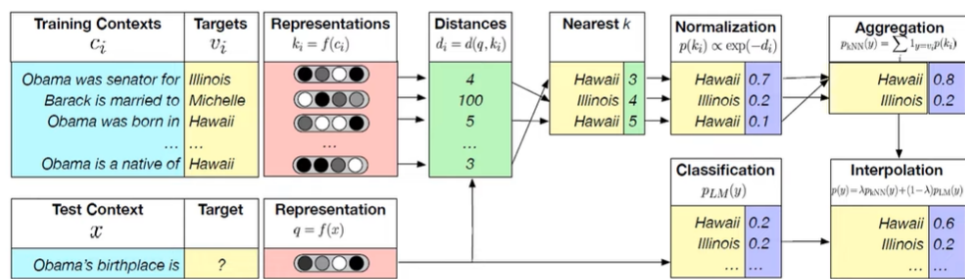
- (다른 decoding 방식과 함께 사용 or softmax 함수 layer를 사용하는 모든 task에서 distribution 조정하고자 사용)



<https://docs.aws.amazon.com/prescriptive-guidance/latest/ml-quantifying-uncertainty/temp-scaling.html>

- Logits 값을 상수 T 로 나누고 softmax의 입력을 사용 (이 자체로는 decoding 알고리즘 x) $T \rightarrow$ hyperparameter
- T 클수록 : diverse/risky outputs, vocab 내 probability 차이가 작아짐
- T 작을수록 : generic/safe outputs, vocab 내 probability 차이가 커짐

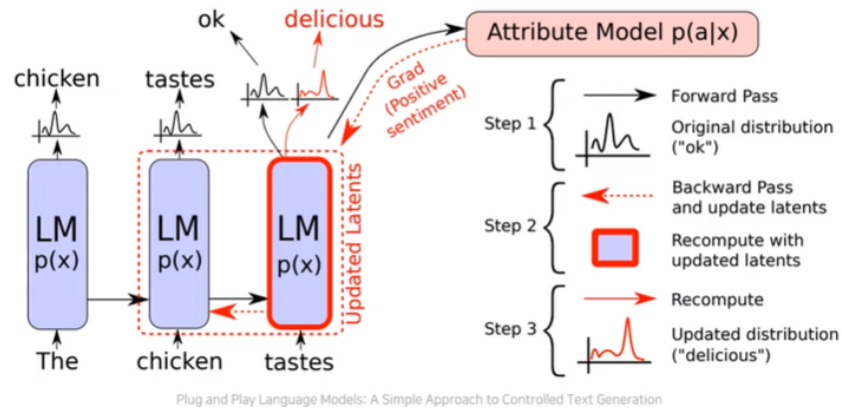
4. Re-balancing distributions : KNN - LM



Generalization through Memorization: Nearest Neighbor Language Models

- training contexts와 targets 학습 \rightarrow 학습된 문장들의 representation과 비교하여 모델의 토큰 distribution을 보정해주는 방식
- K개의 인접한 representation 문장의 target 값을 사용해 모델 평가 결과 보정

6. Re-balancing : PPLM (Plug and Play Language Model 약어)



- 별도의 모델을 사용하여 기존의 Language 모델 성능 향상
- 추가적인 모델을 사용해 언어모델의 distribution 조정 (ex Sentiment, perplexity)
- Attribute model로부터 gradient(positive sentiment)를 전달받아 latent 업데이트 → Model distribution 업데이트 진행
- 연속되는 문장의 hidden representation similarity를 낮추는 방식 - 문장 내 반복 막을 순 없다 !!
- 동일한 단어가 등장하는 것을 막는 attention mechanism : Coverage loss
- 이미 등장한 토큰에 패널티 부여 : Unlikelihood objective

▼ Training

▼ Diversity Issues

▼ Unlikelyhood Training

- 이미 생성된 토큰이 생성될 확률을 낮추는 패널티를 기존 loss function에 추가

$$\mathcal{C} = \{y^*\}_{<t}$$

$$\mathcal{L}_{UL}^t = - \sum_{y_{neg} \in \mathcal{C}} \log(1 - P(y_{neg} | \{y^*\}_{<t}))$$

$$\mathcal{L}_{ULE}^t = \mathcal{L}_{MLE}^t + \alpha \mathcal{L}_{UL}^t$$

Neural Text Generation with Unlikelihood Training

- 반복된 단어, 구의 생성을 줄일 수 있음

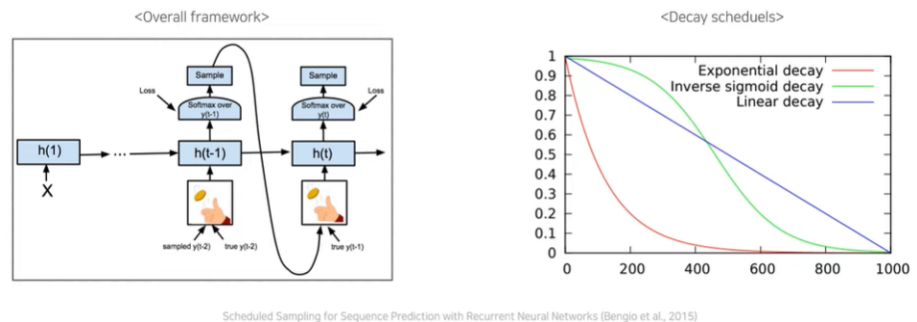
- 생성되는 text의 다양성을 증가시킴

▼ Teacher forcing → Bias

- Teacher forcing : 현 토큰에 대해서 단어가 잘못 생성되었을 때, 다음 토큰을 생성하기 위해 반복해서 들어갔을 때의 문제를 방지하기 위해서 원 문장의 단어를 그대로 사용하는 방식인데, → 이렇게 할시 test와 다르기 때문에 teacher forcing이 학습이 Exposure Bias를 야기할 수 있다고 한다.

▼ Solutions

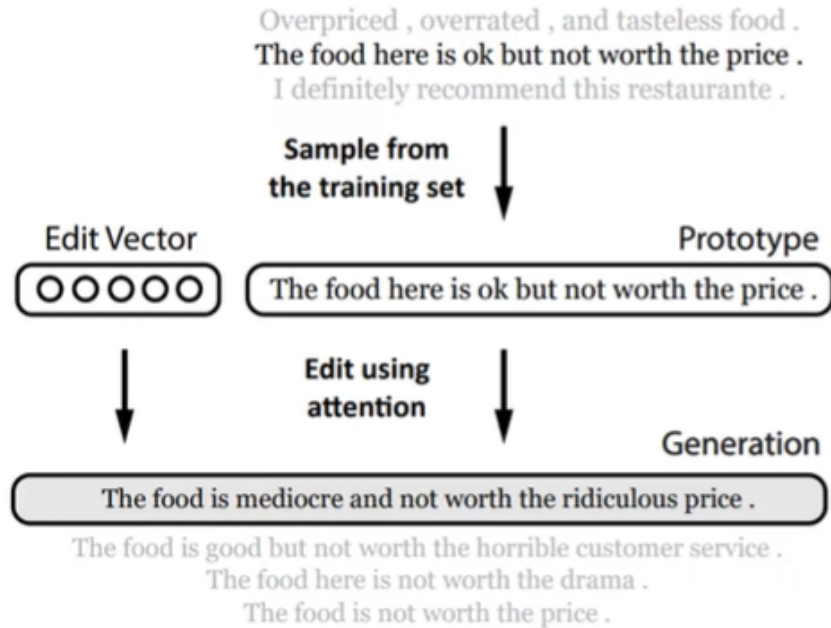
1. Scheduled sampling



- 특정확률로 이전 생성된 토큰을 다음 step의 입력으로 사용
- 학습이 진행될 수록 더 적은 gold token 사용

2. Dataset Aggregation

3. Sequence re-writing



Generating Sentences by Editing Prototypes (Guu, Hashimoto et al., 2018)

- 학습데이터로 구축한 프로토타입 set을 text생성에 사용
- 샘플링한 prototype을 edit vector를 사용해 변형 (Adding, Removing, Modifying tokens로 변형)
- 기존 언어 모델과는 달리 실제 문장을 적절히 변형하는 과정을 통해 → perplexity 등의 성능을 높일 수 있었음

4. Reinforcement Learning

- Text 생성모델을 Markov decision process로 구성
- State : 이전 context의 representation
- Actions : 현재 step 에서 생성될 수 있는 단어
- Policy : Decoder
- REwards : score 함수로부터 받게 될 보상 (ex BLEU, ROUGE, CIDEr, SPIDEr 등)

$$\mathcal{L}_{RL} = - \sum_{t=1}^T r(\hat{y}_t) \log P(\hat{y}_t | \{y^*\}; \{\hat{y}_t\}_{<t})$$

- Reward Estimation
 - 의도하지 않은 shortcut을 모델이 학습하지 않도록 reward function 잘 정의

▼ Evaluation

▼ Content overlap metrics

▼ N-gram overlap metrics : BLEU

$$BLEU = \min(1, \frac{\text{output length}(\text{예측 문장})}{\text{reference length}(\text{실제 문장})}) (\prod_{i=1}^4 \text{precision}_i)^{\frac{1}{4}}$$

짧은 문장에 대한 penalty N-gram precision 의 기하평균

- 실제 문장 대비 짧은 문장을 생성시 패널티 부여
- N-gram precision의 기하평균 사용
- 일반적으로 많이 사용되는 metric

▼ N-gram overlap metrics : ROUGE

$$\text{ROUGE-N} = \frac{\sum_{S \in \{\text{ReferenceSummaries}\}} \sum_{gram_n \in S} \text{Count}_{match}(gram_n)}{\sum_{S \in \{\text{ReferenceSummaries}\}} \sum_{gram_n \in S} \text{Count}(gram_n)}$$

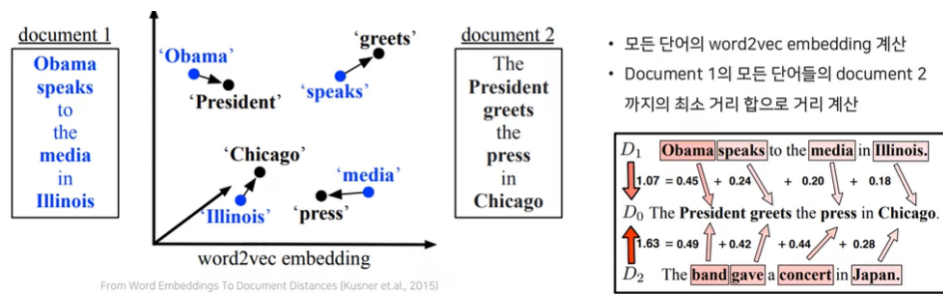
- N - gram recall 사용
- Brevity penalty x
- BLEU와는 달리 n-gram 별로 따로 비교 (ex Rouge-1,2,L)

▼ N-gram overlap matrices 한계

- Summarization output text가 길 때, dialogue task 등 open-ended Machine Translation에서 적절한 지표 x
- 단어의 문맥적 의미 반영 x
- 사람의 평가와 상관성이 낮음

▼ Solution (Semantic overlap metrics, model-based metrics → 비교적 최근에 연구)

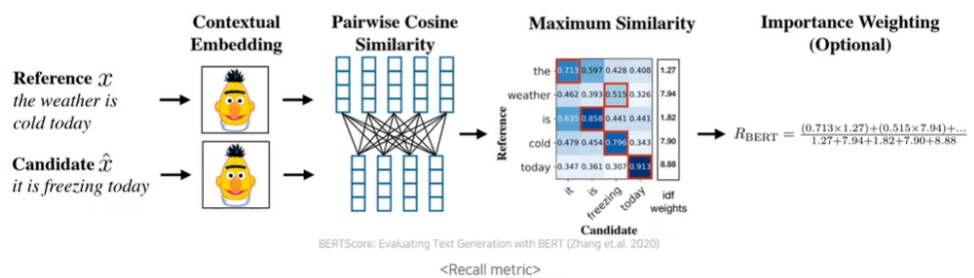
▼ Word Mover's Distance



생성 text와 reference text의 단어 또는 문장의 semantic similarity를 계산할 수 있음

▼ BERTSCORE

사전 학습된 BERT 사용



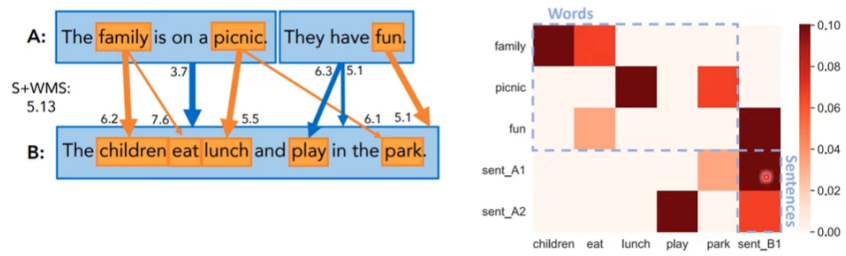
- Reference와 candidate의 contextual embedding 계산
- 모든 pair에 대해 cosine similarity 계산, greedy matching 후 weighted average 구함
- Inverse document frequency score 사용 (the, is 의 경우 단어가 정보를 많이 담고 있지 않아서 idf score 값 낮게 나온다 !!)

$$\text{idf}(w) = -\log \frac{1}{M} \sum_{i=1}^M \mathbb{I}[w \in x^{(i)}]$$

▼ Sentence Movers Similarity

- Sentence level, word level에서 모두 similarity 계산 !!
- stopwords를 제외한 모든 단어가 pair로 계산된다
- sentence embedding, word embedding의 similarity 모두 계산

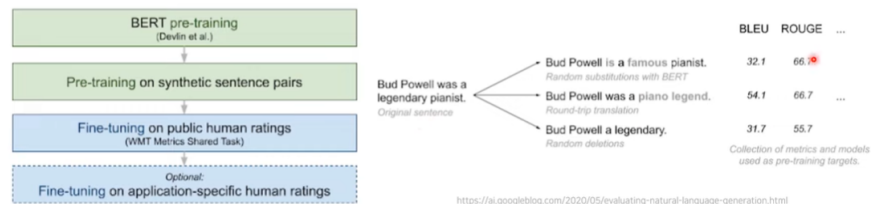
→ sentence embedding 값은 그 문장에서 구성하고 있는 모든 word들의 embedding 값의 합으로 구성된다 !!



Sentence Mover's Similarity : Automatic Evaluation for Multi-Sentence Texts (Clark et al., 2019)

▼ BLEURT

- 문장 유사도를 예측하는 BERT 기반의 regression model을 학습한다 !!!



<https://ai.googleblog.com/2020/05/evaluating-natural-language-generation.html>

BLEURT: Learning Robust Metrics for Text Generation (Sellam et al. 2020)

▼ ETC

- Ethical considerations
- Large scale 언어모델의 활용 가능성