Language model interpolation

- unseen word sequence에 대처하는 방법
- 다른 Language Model을 linear하게 일정비율로 섞는 것

$$\tilde{P}(w_n | w_{n-k}, \cdots, w_{n-1}) = \lambda P_1(w_n | w_{n-k}, \cdots, w_{n-1}) + (1-\lambda) P_2(w_n | w_{n-k} \cdots w_{n-1})$$

(P1, P2: Language model)

☆그냥 모든 corpus를 합쳐서 만들지 않는 이유

Domain specific corpus의 양이 적어서 발생 안될 가능성↑

적절한 Interpolation ratio(λ)를 찾는 것 중요

Back off

n-gram의 n을 줄여가면서 확률을 구하는 것 ex) Trigram → Bi-gram → Uni-gram

$$\tilde{P}(w_n | w_{n-k} \cdots w_{n-1}) = \lambda_1 P(w_n | w_{n-k} \cdots w_{n-1}) + \lambda_2 P(w_n | w_{n-k+1} \cdots w_{n-1}) + \cdots + \lambda_k P(w_n),$$

where $\sum_i \lambda_i = 1$.

Summary

전통적인 NLP는 단어를 discrete symbol로 보기 때문에 문제 발생

- Exact matching에 대해서만 count를 하며, 확률도 approximation

ex) 분홍과 핑크의 의미나 유사한지 신경X 서로 아예 다른 것

Perplexity (PPL)

- Test set에 대해서 언어모델을 이용하여 확률(Likelihood)을 구하고 PPL 수식을 통해 성능 측정

$$PPL(x_1, \cdots, x_n; \theta) = P(x_1, \cdots, x_n; \theta)^{-\frac{1}{n}} = \sqrt[n]{\frac{1}{P(x_1, \cdots x_n; \theta)}}$$

PPL이 작을수록 좋은 언어모델

Perplexity

- Timestep 별 평균 branch의 수
- 즉 매 PPL 마다 timestep 마다 가능한 가짓수인 6

PPL이 ↓ 확률분포가 sharp 하다.

PPL이 ↑ 확률분포가 flat 하다.

언어모델의 정의

- 낮을수록 Good
- 확률의 역수에 문장길이로 기하 평균
- 매 time step 마다 평균적으로 헷갈리고(no clue 완전히) 있는 단어의 수

N-gram

장점: · 쉽게 대형 시스템에 적용가능
· N-gram 훈련 및 측정 방식이 굉장히 간편

단점
· unseen word sequence에 대해 어려움
· 멀리 있는 단어에 대해 대처 불가
ex) Markov Assumption
· N이 커질수록 용량도 커짐

Auto Regressive Task

- NLG
- 이전 time step의 출력에 따라 현재 입력의 state가 바뀌는 것 (이전의 출력이 앞으로의 것에 영향을 끼치는 것)

So

- 학습시에는 이전 정답의 출력이 아닌 정답을 RNN에 넣어줌

Problem

1. inference를 위한 code 필요
2. 학습과 추론 방법의 괴리로 발생하여 성능 저하

transformer의 동작 이유?