Teaching Machines to Read and Comprehend (15/6/10)

Before we go on..

1. 발표하면서 여러분께 더 많이 배워가는것이 목표!
    - 빈슬라이드 / 제가 여러분께 드릴 질문이 많습니다. 도와주세요

2. 디자인? 이쁜폰트? 사이즈통일? 그런거 전~혀없습니다. 편하게 봐주세요

3. 손으로 그린 그림.. 이 많은데 잘그린 그림이 아닙니다.

논문 리딩 전에 :

1. MRC란

2. NLU/CM?

3. 그럼 내가 발제할 CNN/Dailymail = 이게 왜필요?

4. 그럼 MRC 어떻게 측정? ⇒ 측정척도종류 & 장단점

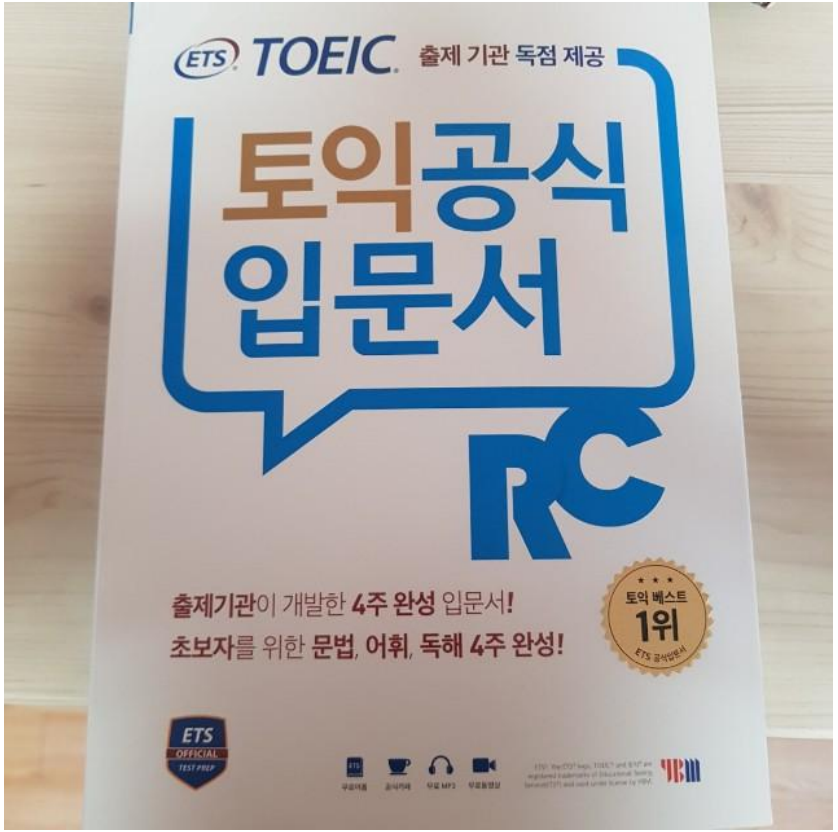
5. MRC의 처리 수행과정

1. MRC란?

~을 이야기하기전에, RC란?

1. MRC란?

~을 이야기하기전에, RC란?



1. MRC란?

Machine Reading Comprehension

- 기계가 글과 질문을 읽고
- 추론을 하여
- 글에서 정답을 찾아주는 것

1. MRC 의 친구들

MRC = CM과 함께 NLU의 끝판왕

(자연어 이해) (대화모델) (질의응답모델)

- 주어진 문서를 읽고(R) 이해(C)한 후 질문에 대한 정답을 찾아내기
    1. 주어진 지문(Context)을 학습하고 (이해)
    2. 질의(Query)에 대한 답변을 추론하는 기술

출처 : 애국기업 LG CNS 블로그

1. MRC란?

차이점을 기억해두세요! 이따 논문에서 언급합니다.

Difference from other QA settings

- **General QA**: use everything
  - rules/templates, knowledge bases/dictionaries, raw text, ...
- **Reading comprehension**: a single knowledge source
  - forces the model to “comprehend”

1. MRC란? - 예시

MRC가 잘 활용될 수 있는 분야 중 하나는 질의응답(QA/Question Answering) 시스템이며,

- 마이크로소프트가 인수한 인공지능 기업 말루바(Maluuba)는 2016년 3월,
- 소설 **'해리포터와 마법사의 돌'**에 대한 무작위 질문에 대답할 수 있는 Machine Reading 시스템을 개발하여 선보인 바 있습니다.
- 말루바는 현재 자동차, 가전기기 등의 **사용 설명서**를 이해하고 사용자의 질문에 대답할 수 있는 시스템의 개발을 진행 중입니다.

1.5 CM?

- CM : Conversation Model

- 입력 문장을 이해. 답변 생성. 대화 흐름관리
- 1.대화 흐름과 2. 문맥을 고려하여,
- 사용자의 발화 의도에 대한 최선의 대화 전략을 결정하고,
- 시스템이 다음으로 발화할 의미 Feature를 생성합니다.

3. 그럼 딥마인드는 이걸 왜만들었음?

이따 논문에 나오지만, 데이터셋 (QDA트리플) 이 절대적으로 부족

기계가 다양한 주제의 글을 읽어서 뜻을 이해하고 (MRC: Machine Reading Comprehension), 읽은 글에 대한 이해력을 평가하기 위해 질의응답 기술 개발을 위해 질문-정답-정답단락의 트리플로 구성된 대규모의 데이터셋이 공개되고 있다. 이미 이미지넷(ImageNet)

스탠포드 대학 뿐만 아니라 IBM, 구글, 마이크로소프트, 페이스북 등의 글로벌 기업들도 질문에 포함된 단어 및 단어들의 어순을 매칭하여 정답을 추론하는 기존의 질의응답 기술에서 탈피하기 위해, 기계가 글을 읽어서 뜻을 이해하는 능력인 독해력(Reading Comprehension)과 읽은 글의 이해정도를 평가하기 위해 질의응답(Question Answering) 기술을 딥러닝 기반으로 개발하고 있다. 어휘와 문장의 문법분석에서 나

4. MRC 측정척도들의 종류 & 장단점

참고 : SQuAD는 1.0 버젼 기준 설명 ; 현재는 2.0 (차이는 몇슬라이드 뒤에)

표 1 RC/QA 데이터셋

데이터셋	질문 구축	질문 수	문서 수 (문서유형)	URL
CNN/ Daily Mail	빈칸 메우기 질문 생성	387,420 (CNN 뉴스) 997,467 (Daily Mail 뉴스)	92,579 (CNN 뉴스) 219,506 (Daily Mail 뉴스)	http://cs.nyu.edu/~kcho/DMQA/
SQuAD	크라우드 소싱	107,785	536 (위키피디아)	https://rajpurkar.github.io/SQuAD-explorer/
MS MARCO	빙 검색로그	100,000	1M 정답단락, 200,000+ 웹문서 링크	http://www.msmarco.org/
NewsQA	크라우드 소싱	119,633	12,744건 (CNN 뉴스)	https://datasets.maluuba.com/NewsQA

4.1 좀 옛날거네.. 장단점?

1.양으로 승부

2.변별력 안좋음

=> 그닥..

- **CNN/Daily Mail(Google DeepMind & University of Oxford, 2015):** 구글 딥마인드는 실세계의 독해력 문제를 반영하는 대용량 데이터셋 구축을 목표로, 미국의 CNN 뉴스기사를 2007년 4월부터 2015년 4월까지, 영국 Daily Mail의 뉴스기사를 2010년 6월부터 2015년 4월까지 각각 수집하여, 정답이 들어있는 문서셋 312,085건을 구축함. 원본문서에서 나타나는 엔티티들을 익명화하여 변환 후, "X"로 명기된 엔티티(단어 또는 구/절)를 찾아야 하는 질문셋 1,384,887건을 구축함[1]. 그러나 이 데이터셋은 에러가 많이 포함되어 있다는 지적이 있으며, 엔티티에 대한 빈칸 채우기 문제(Cloze type question)는 독해력 난이도가 낮아 사람의 성능과 근접한 연구결과가 도출되고 있어 기계의 독해력 판별에는 변별력이 낮다는 지적이 제기되고 있음[2]

4.2. SQuAD?

= The Stanford Question Answering Dataset

= 컴퓨터가 사람처럼 주어진 문서를 읽고 이해한 후 질문에 대한 정답을 찾아내는 MRC 테스트의 하나

- **SQuAD(Stanford Question Answering Dataset, 2016):** 스탠포드 대학에서 위키피디아 문서 536건을 대상으로 107,785건의 질문을 크라우드 소싱을 통해 만듬. 정답을 추출하는데 제공되는 정답단락은 4-5개의 문장으로 구성되어서 QA문제로는 정답단락이 너무 짧다는 문제점이 제기됨. 이와 같은 문제점으로 인해, 인간의 정답율이 91.2%(F1)인데 비해 최고성능이 84%(F1)로 매우 높음(2017년 5월 기준)[3]

4.2. SQuAD !

- 1.0의 문제 : 정답률이 너무높다.
  - (84%, 인간은 91%)
- 주어진 문장내에 답이 없는 문제들도 만들자!
  - => SQuAD 2.0
  - = 해답(A)이 지문(C) 내에 없는 질문(Q)도 존재
- 그래서 지금은?

4.2. SQuAD !

- 1.0의 문제 : 정답률이 너무높다.
  - (84%, 인간은 91%)
- 주어진 문장내에 답이 없는 문제들도 만들자!
  - => SQuAD 2.0
  - = 해답(A)이 지문(C) 내에 없는 질문(Q) 존재
- 그래서 지금은?
- => BERT 짱

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar & Jia et al. '18)	86.831	89.452
1 Mar 05, 2019	BERT + N-Gram Masking + Synthetic Self-Training (ensemble) *Google AI Language* https://github.com/google-research/bert	**86.673**	**89.147**
2 Mar 05, 2019	BERT + N-Gram Masking + Synthetic Self-Training (single model) *Google AI Language* https://github.com/google-research/bert	85.150	87.715
3 Jan 15, 2019	BERT + MMFT + ADA (ensemble) *Microsoft Research Asia*	85.082	87.615
4 Jan 10, 2019	BERT + Synthetic Self-Training (ensemble) *Google AI Language* https://github.com/google-research/bert	84.292	86.967
5 Dec 16, 2018	Lunet + Verifier + BERT (ensemble) *Layer 6 AI NLP Team*	83.469	86.043
5 Dec 21, 2018	PAML+BERT (ensemble model) *PINGAN GammaLab*	83.457	86.122
5 Dec 13, 2018	BERT finetune baseline (ensemble) *Anonymous*	83.536	86.096

4.2. SQuAD !

- 발표자료 만들때 순위 뒤집힘 + 휴먼퍼포먼스 넘어섬 !
- BERT+DAE+AoA

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar & Jia et al. '18)	86.831	89.452
1 Mar 20, 2019	BERT + DAE + AoA (ensemble) *Joint Laboratory of HIT and iFLYTEK Research*	**87.147**	**89.474**
2 Mar 15, 2019	BERT + ConvLSTM + MTL + Verifier (ensemble) *Layer 6 AI*	86.730	89.286
3 Mar 05, 2019	BERT + N-Gram Masking + Synthetic Self-Training (ensemble) *Google AI Language* https://github.com/google-research/bert	86.673	89.147
4 Mar 16, 2019	BERT + DAE + AoA (single model) *Joint Laboratory of HIT and iFLYTEK Research*	85.884	88.621
5 Jan 15, 2019	BERT + MMFT + ADA (ensemble) *Microsoft Research Asia*	85.082	87.615
5 Mar 13, 2019	BERT + ConvLSTM + MTL + Verifier (single model) *Layer 6 AI*	84.924	88.204
5 Mar 05, 2019	BERT + N-Gram Masking + Synthetic Self-Training (single model) *Google AI Language* https://github.com/google-research/bert	85.150	87.715
6 Jan 10, 2019	BERT + Synthetic Self-Training (ensemble) *Google AI Language* https://github.com/google-research/bert	84.292	86.967

5. MRC의 처리수행과정

- 1. 이해 ● Context & Query에 대한 이해
- 2. 관계 파악● Context와 Query의 관계 파악
- 3. 답변 검색● Context로부터 답변 검색 ⇒ 제공

출처 : 애국기업 LG CNS 블로그

논문 리뷰 시작 !

0. Abstract

Abstract

Teaching machines to read natural language documents remains an elusive challenge. Machine reading systems can be tested on their ability to answer questions posed on the contents of documents that they have seen, but until now large scale training and test datasets have been missing for this type of evaluation. In this work we define a new methodology that resolves this bottleneck and provides large scale supervised reading comprehension data. This allows us to develop a class of attention based deep neural networks that learn to read real documents and answer complex questions with minimal prior knowledge of language structure.

MRC 분야는 대규모 지도 학습 데이터셋 부재가 큰 문제

-> Attention + RNN으로 해결

1. Introduction

1. MRC 성능평가를 하고싶다.
2. But 감독 된 자연어 읽기 독해 데이터를 얻는 것이 어렵다
3. 계산언어학에서 많이 해결 시도했으나 ; 문제(2)
    - 1. 소량
    - 2. 일반화 능력 저하 :
        - “..그러나 역사적으로 계산 언어학에서의 많은 유사한 접근법은 합성 된 자료에서 실제 환경으로의 전환을 관리하지 못했다. 왜냐하면 폐쇄 된 세계는 필연적으로 **자연어의 복잡성, 풍부함 및 소음을 포착하지 못하기 때문이다..**”

4. 그래서 딥마인드가 나섰다

1. Introduction

- 목표 : 대용량의 문맥 - 쿼리 - 응답 트리플 (C-Q-A triple) 생성
- 방법 :
  - 1. 개체 익명화
  - 2 . 빈칸채우기 (클로제테스트 = 필인더블랭크)
- 장점 :
  - 1. 기계 스스로 (주어진 질문에 대답하는 데 도움이 될 것으로 생각되는) 문서의 측면에 스스로 초점을 맞출 수 있으며, (LSTM/어텐션)
  - 2. 또한 추론 프로세스를 시각화 할 수 있습니다. (뒤에나옵니다. 근데 별거없음..)

막간: 재미로보는 구글 vs 파파고(1)

원문 :

We compare these neural models to a range of baselines and heuristic benchmarks based upon a traditional frame semantic analysis provided by a state-of-the-art natural language processing (NLP) pipeline.

파파고 :

우리는 이러한 신경 모델을 최첨단 자연 언어 처리(NLP) 파이프라인이 제공하는 전통적인 프레임 의미 분석에 기초한 베이스라인 및 휴리스틱 벤치마크의 범위에 비교한다.

-> 읽어도 뭔소린지 모르겠음

구글 :

최첨단 자연 언어 처리 (NLP) 파이프 라인이 제공하는 전통적인 프레임 의미 분석을 기반으로 다양한 신경 모델을 다양한 기준선 및 휴리스틱 벤치 마크와 비교합니다.

-> 구글승 (필요한 구 가 앞으로 배치되어있음)

2. supervised training data for reading comprehension

목표 : 대용량의 p(a|c,q) 생성

- = p(answer | context_document, query)

2. supervised training data for reading comprehension

제 질문) 코레퍼런스가 뭐죠? => (토막상식) co-reference (resolution)

Def ; Identify all noun phrases (mentions) that refer to the same real world entity =같은 실재를 나타내는 모든 명사구들을 찾아내는 작업

= 즉, NE 중에 → 명사구 중에 → 동일실체 point하는애들 찾기

Anaphors(전방조응사) , cataphors (는 신경쓸필요x)

! Anaphor 가 coreference 에 포함됨

2.1 어떻게 만드나?

Step1. 말뭉치(corpus) 생성

1. ㄱ.온라인 신문 기사 ㄴ.그 기사 요약본 => 2 말뭉치 생성
2. CNN에서 93k개 기사, Daily Mail 웹 사이트에서 220k개 기사 수집
3. 기사에 포함 된 정보의 측면을 요약
4. 핵심 : 요약본은 단순히 원 기사에서 문장을 복사하지 않습니다.

Step2. ACQ트리플 만들기

- 기사요약본들을 Cloze 스타일 질문으로 변환 = 문서-질의-응답 트리플 말뭉치 생성

결과 : 약 120만개 코퍼스

2.1 entity replacement & permutation(치환/순서바꿈)

논문 핵심 : 모델이 세계 지식이나 동시 발생 능력이 아닌,

"오직 주어진 문서만" 읽고, 이해하고, 대답하는 능력을 평가할수있는 셋을 만드는것.

- => 예) 수능 언어영역 문제에 비교시
- 1.공부 전~혀안해도 상식만있으면 풀수있던문제
- 2.지문을 완벽히 이해못하면 못푸는 문제

how?

1. Cloze style (fill in the blank)
2. Anonymisation

2. supervised training data for RC

- a) 각 데이터 포인트에서 핵심어를 확립하기 위해 coreference 시스템을 사용한다.
- b) 모든 entity를 coreference에 따라 추상적 엔티티 마커로 대체
- c) 데이터 요소(데이터포인트) 가 로드 될 때마다 이러한 엔티티 마커를 무작위로 바꿉니다.

Original Version	Anonymised Version
Context	
The BBC producer allegedly struck by Jeremy Clarkson will not press charges against the "Top Gear" host, his lawyer said Friday. Clarkson, who hosted one of the most-watched television shows in the world, was dropped by the BBC Wednesday after an internal investigation by the British broadcaster found he had subjected producer Oisin Tymon "to an unprovoked physical and verbal attack." ...	the *ent381* producer allegedly struck by *ent212* will not press charges against the " *ent153* " host , his lawyer said friday . *ent212* , who hosted one of the most - watched television shows in the world , was dropped by the *ent381* wednesday after an internal investigation by the *ent180* broadcaster found he had subjected producer *ent193* " to an unprovoked physical and verbal attack . " ...
Query	
Producer **X** will not press charges against Jeremy Clarkson, his lawyer says.	producer **X** will not press charges against *ent212* , his lawyer says .
Answer	
Oisin Tymon	*ent193*

Table 3: Original and anonymised version of a data point from the Daily Mail validation set. The anonymised entity markers are constantly permuted during training and testing.

재미로보는 구글 vs 파파고(2)

원문) The hi-tech bra that helps you beat breast X;

재미로보는 구글 vs 파파고(2)

원문) The hi-tech bra that helps you beat breast X;

1. 구글) 가슴 ( ) 를 이길 수있는 하이테크 브래지어;

재미로보는 구글 vs 파파고(2)

원문) The hi-tech bra that helps you beat breast X;

1. 구글) 가슴 ( ) 를 이길 수있는 하이테크 브래지어;
2. 파파고) 가슴 ( ) 를 때리는 데 도움을 주는 하이테크 브라;

재미로보는 구글 vs 파파고(2)

원문) The hi-tech bra that helps you beat breast X;

1. 구글) 가슴 ( ) 를 이길 수있는 하이테크 브래지어;
2. 파파고) 가슴 ( ) 를 때리는 데 도움을 주는 하이테크 브라;

=> 구글승 (2:0),

But 파파고의 “브라"라는 좀 더 토속적인 단어를 이용한점은 굿!

3. Models ; 여기 너무 어려움..

제 질문)

?frame semantic annotation?

?entity-predicate triple? ; 개체-술어 ?

?semantic triple?

?RDF triple?

?prop bank triple?

3. Models ; 여기 너무 어려움..

제 질문)

!frame :

- ≈ 문장 구조로 나눈 종류(?)클래스(?)
  - like 성문기초영문법에서 여러 문장종류들 :
  - 1.주어동사, 2.주어타동사목적어 3. .. 등등
- 모든 문장이 프레임?이 있다.
  - 예) 내가 달리다.
  - ~. 라는 문장은, 목적어 없어도 되는 프레임.
  - 근데 어떤 문장은, 사용된 동사는 반드시 목적어가. 있어야하고.
- = 구조에 따라 모든 문장들은 특정 프레임으로 분류 가능
-

3.1 Symbolic Matching Models (기호매칭모델)

NULL

3.1.1. Frame-Semantic Parsing

?명확성을 위해 모든 PropBank 트리플이 (e1, V, e2) 형식 인 것으로 가정합니다.?

아 6개 각각이 전략이고.. 우선순위에 의해 정렬되어 문제해결..?

	Strategy	Pattern $\in q$	Pattern $\in d$	Example (Cloze / Context)
1	Exact match	(p, V, y)	$(\boldsymbol{x}, V, y)$	X loves Suse / **Kim** loves Suse
2	be.01.V match	$(p, be.01.V, y)$	$(\boldsymbol{x}, be.01.V, y)$	X is president / **Mike** is president
3	Correct frame	(p, V, y)	$(\boldsymbol{x}, V, z)$	X won Oscar / **Tom** won Academy Award
4	Permuted frame	(p, V, y)	$(y, V, \boldsymbol{x})$	X met Suse / Suse met **Tom**
5	Matching entity	(p, V, y)	$(\boldsymbol{x}, Z, y)$	X likes candy / **Tom** loves candy
6	Back-off strategy	*Pick the most frequent entity from the context that doesn't appear in the query*		

2번은 V가 be동사일때고,1,3 차이점? / 4,5,6은 왜, 무슨목적으로 존재하는지 모르겠다..

Table 4: Resolution strategies using PropBank triples. $\boldsymbol{x}$ denotes the entity proposed as answer, V is a fully qualified PropBank frame (e.g. *give.01.V*). Strategies are ordered by precedence and answers determined accordingly. This heuristic algorithm was iteratively tuned on the validation data set.

3.1.2 Word Distance Benchmark

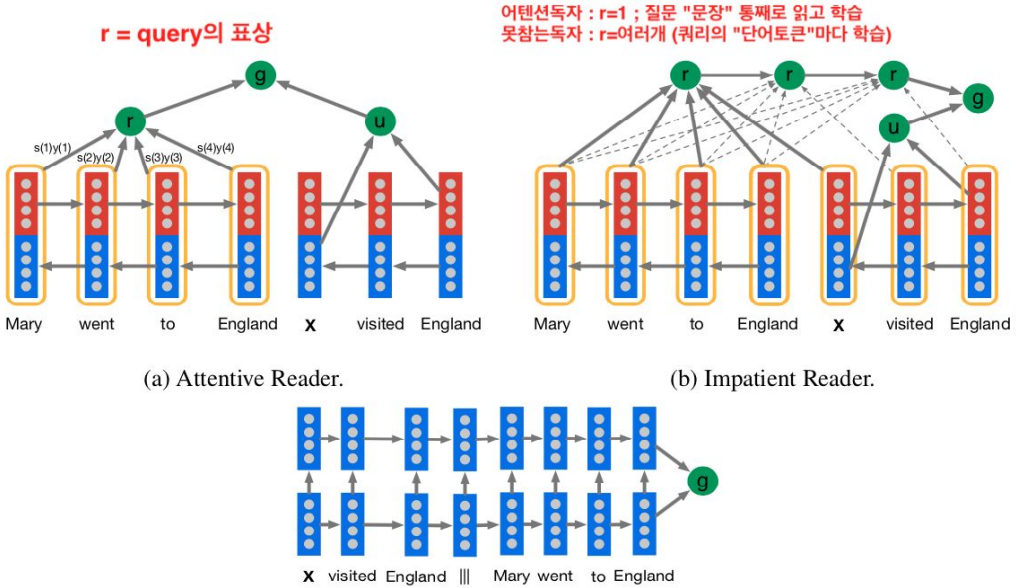
?정말 모르겠음

번역)

- 우리는 단어 거리 측정에 의존하는 또 다른 기준을 고려합니다.
- 여기에서는 Cloze 형식의 질문에서의 "자리 표시자(빈칸, 맞춰야할놈)을, 컨텍스트 문서에서 (답변이 되는것이 가능한) 각 엔터티와 얼라인 시킨 뒤, (정렬 된 엔터티 주변에서) 질문과 컨텍스트 간의 거리 메져를 계산합니다.
- 이 점수는 q에있는 모든 단어각각에 대해 가장 가까운 d의 단어와의 거리를 합산하여 계산. 정렬은 단어를 직접 일치 시키거나 코레퍼런스 시스템으로 정렬하여 정의합니다.
- 유효성 확인 데이터에서 단어 당 최대 페널티 (m = 8)를 조정합니다.

3.2. Neural Network Models

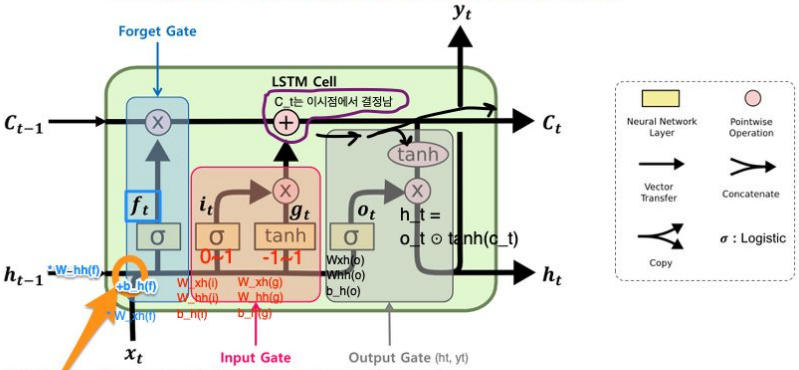
1. Deep LSTM Reader
2. Attention Reader
3. Impatient Reader (Attention 변형)



3.2.1 LSTM?

:Pass

깨달음:
- 시그모이드 = 그냥 온오프 스위치
- 하이퍼탄젠트 = 실제연산 (RNN,GRU 모두 얘가 핵심임)



인풋/아웃풋 게이트 그림이 복잡해보이는 이유는 h_t-1이랑 x_t는 "concate"해서 한줄기로 흘러가는걸 그렸기때문 (실제연산시처럼)

$f_t = \sigma(W_{xh(f)} \cdot x_t + W_{hh(f)} \cdot h_{t-1} + b_{h(f)})$

$i_t = \text{sigmoid}(W_{xh(i)} \cdot x_t + W_{hh(i)} \cdot h_{t-1} + b_{h(i)})$

$g_t = \tanh(W_{xh(g)} \cdot X_t + W_{hh(g)} \cdot h_{t-1} + b_{h(g)})$

$O_t = \sigma(W_{xh(o)} \cdot X_t + W_{hh(o)} \cdot H_{t-1} + b_{h(o)})$

최종수식
1. 셀스테이트 - 기억선별기능 : $C_t = F_t \odot C_{t-1} + I_t \odot G_t$
2. 히든스테이트 - 피쳐 리프레젠~ : $H_t = O_t \odot \tanh(C_t)$

W_xhf,Whhf
= 이번 인풋들(Ht-1&Xt)을 토대로 계산하여, 기존 Ct-1 각 요소값들을 얼마나 지워낼까요?
= 기존의 기억들을 얼마나 "잊어야" 하나요?

<INPUT GATE>

Wxhi,Whhi
= 이번 인풋들의 - 각각의 스칼라값들을, (오른쪽의 g_t값에 대해)
0~1 사이에서 얼마나 어떤 element들을 업데이트해줘야 로스 감소?~를 학습

<UPDATE GATE>

Wxhg,Whhg
= 이번 인풋들을 가지고 / 실제 RNN연산= Cell state에 더해질 "후보값(C~t) 을 tanh로 생성
⟶ 추후 이값에 i_t가 elementwise하게 각 요소 스칼라값들을 0~1 사이로 조절해줌

<output gate>

Wxho, Whho = 이번 인풋들을 토대로, 시그모이드를 이용, 스위치를 만든다음 ⟶ 새로 업데이트된 (실제 RNN연산인 tanh를 C_t에 한 결과인) 후보값 C~t 의 각 엘레멘트값들을 0~1 사이로 조절

3.2.1. Deep-LSTM Reader

모델 상세설명 :

1. 다큐먼트 전체를, 워드 하나씩 피드후 쿼리를 넣음
   a. 반대로도 함 (쿼리먼저 ; 다큐먼트는 그다음)
2. 결과 : 모델은 질문-문서 페어를 하나의 긴 문장처럼 프로세싱 -> 표상화.
3. 그 임베딩을 토대로, 문단내 어떤 토큰이 쿼리를 대답하는지 예측

모델 특징 :

1. Multi layer
2. peephole
3. skip-connection

3.2.1. Deep-LSTM Reader

아름다운수식들 ; 핍홀부터 까면 친숙한 lstm모양이되니까 핍홀부터.

$$
\begin{aligned}
x'(t,k) &= x(t)||y'(t,k-1), \qquad y(t) = y'(t,1)||\ldots||y'(t,K) \\
i(t,k) &= \sigma\left(W_{kxi}x'(t,k) + W_{khi}h(t-1,k) + W_{kci}c(t-1,k) + b_{ki}\right) \\
f(t,k) &= \sigma\left(W_{kxf}x(t) + W_{khf}h(t-1,k) + W_{kcf}c(t-1,k) + b_{kf}\right) \\
c(t,k) &= f(t,k)c(t-1,k) + i(t,k)\tanh\left(W_{kxc}x'(t,k) + W_{khc}h(t-1,k) + b_{kc}\right) \\
o(t,k) &= \sigma\left(W_{kxo}x'(t,k) + W_{kho}h(t-1,k) + W_{kco}c(t,k) + b_{ko}\right) \\
h(t,k) &= o(t,k)\tanh\left(c(t,k)\right) \\
y'(t,k) &= W_{ky}h(t,k) + b_{ky}
\end{aligned}
$$

3.2.1.1.peephole?

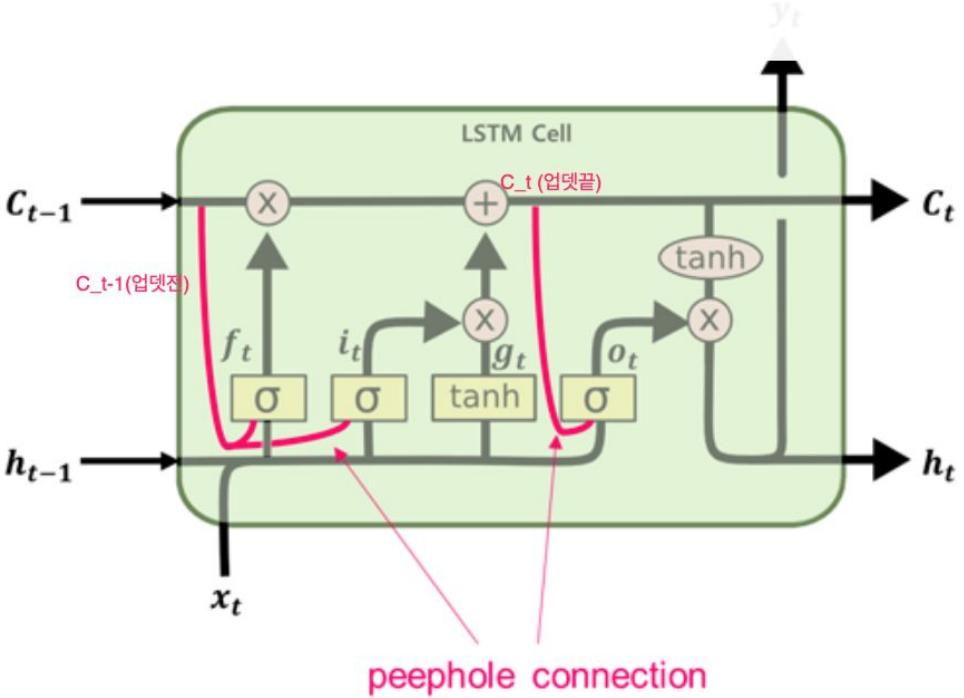
이런 설명도 있지만..

또 다른 확장으로 핍홀peephole 기능이 있다[Gers2000b]. [그림 8-19]는 핍홀을 추가한 LSTM이다. 핍홀은 메모리 블록의 내부 상태를 3개의 게이트에 알려 주는 역할을 한다. 노란색의 에지가 핍홀 기능을 한다. 순차 데이터를 처리하다가 어떤 조건에 따라 특별한 조치를 취해야 하는 응용 문제에서는 핍홀 기능이 매우 효과적이다. 예를 들어, 입력 문장에서 특정 단어가 3번 이상 나타나면 이후에 들어 올 입력을 무시하고 특정한 메시지를 내보내야 하는 응용을 들 수 있다. 또는 음성 인식을 수행하다가 특정한 단어가 발견되면 지정된 행위를 하는 응용도 이에 해당한다.

Peephole 한방에 이해하기 (1)

그림이 개똥같안서 그렇지 알고나면 정말 별거아님

+ 변태적인 이름



Peephole 한방에 이해하기 (2)

Peephole 한방에 이해하기 (3)

LSTM

$$\mathbf{f}_t = \sigma\left(\mathbf{W}_{xf}^T \cdot \mathbf{x}_t + \mathbf{W}_{hf}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_f \right)$$

$$\mathbf{i}_t = \sigma\left(\mathbf{W}_{xi}^T \cdot \mathbf{x}_t + \mathbf{W}_{hi}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_i \right)$$

$$\mathbf{o}_t = \sigma\left(\mathbf{W}_{xo}^T \cdot \mathbf{x}_t + \mathbf{W}_{ho}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_o \right)$$

Peephole 한방에 이해하기 (4)

LSTM + peephole

LSTM에 비해 추가된부분

$$\mathbf{f}_t = \sigma\left(\mathbf{W}_{cf}^T \cdot \mathbf{c}_{t-1} + \mathbf{W}_{xf}^T \cdot \mathbf{x}_t + \mathbf{W}_{hf}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_f\right)$$

$$\mathbf{i}_t = \sigma\left(\mathbf{W}_{ci}^T \cdot \mathbf{c}_{t-1} + \mathbf{W}_{xi}^T \cdot \mathbf{x}_t + \mathbf{W}_{hi}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_i\right)$$

$$\mathbf{o}_t = \sigma\left(\mathbf{W}_{co}^T \cdot \mathbf{c}_t + \mathbf{W}_{xo}^T \cdot \mathbf{x}_t + \mathbf{W}_{ho}^T \cdot \mathbf{h}_{t-1} + \mathbf{b}_o\right)$$

3.2.1. Deep-LSTM Reader

아름다운수식 -> 알고보면 정~~~말 별거아님 황당할정도

좀 더 쉬운 설명을 위해, 아래 순서대로 수식을 재정비후 설명하겠음

1. W 아래첨자 제거
    a. 참고 : W_{LIG} : Layer, Input, Gate
2. 연산순서 이해쉽도록 수식순서 재편
3. 손그림과 함께!

3.2.1. Deep-LSTM Reader

원 수식

$$
\begin{aligned}
x'(t,k) &= x(t)||y'(t,k-1), \qquad y(t) = y'(t,1)||\ldots||y'(t,K) \\
i(t,k) &= \sigma\left(W_{kxi}x'(t,k) + W_{khi}h(t-1,k) + W_{kci}c(t-1,k) + b_{ki}\right) \\
f(t,k) &= \sigma\left(W_{kxf}x(t) + W_{khf}h(t-1,k) + W_{kcf}c(t-1,k) + b_{kf}\right) \\
c(t,k) &= f(t,k)c(t-1,k) + i(t,k)\tanh\left(W_{kxc}x'(t,k) + W_{khc}h(t-1,k) + b_{kc}\right) \\
o(t,k) &= \sigma\left(W_{kxo}x'(t,k) + W_{kho}h(t-1,k) + W_{kco}c(t,k) + b_{ko}\right) \\
h(t,k) &= o(t,k)\tanh\left(c(t,k)\right) \\
y'(t,k) &= W_{ky}h(t,k) + b_{ky}
\end{aligned}
$$

3.2.1. Deep-LSTM Reader

학습 파라미터들의 아래 첨자 제거 (t,k도 지우고 싶었지만)

$$
\begin{aligned}
x'(t,k) &= x(t)||y'(t,k-1), \qquad y(t) = y'(t,1)||\ldots||y'(t,K) \\
i(t,k) &= \sigma\,(W \quad x'(t,k) + W \quad h(t-1,k) + W \quad c(t-1,k) + b \quad) \\
f(t,k) &= \sigma\,(W \quad x(t) + W \quad h(t-1,k) + W \quad c(t-1,k) + b \quad) \\
c(t,k) &= f(t,k)c(t-1,k) + i(t,k)\tanh\,(W \quad x'(t,k) + W \quad h(t-1,k) + b \quad) \\
o(t,k) &= \sigma\,(W \quad x'(t,k) + W \quad h(t-1,k) + W \quad c(t,k) + b \quad) \\
h(t,k) &= o(t,k)\tanh\,(c(t,k)) \\
y'(t,k) &= W \quad h(t,k) + b
\end{aligned}
$$

3.2.1. Deep-LSTM Reader

설명할 순서대로 리어레인지

$$x'(t,k) = x(t)||y'(t,k-1), \qquad y(t) = y'(t,1)||\dots||y'(t,K)$$

$$f(t,k) = \sigma\,(W \quad x(t) + W \quad h(t-1,k) + W \quad c(t-1,k) + b \quad)$$

$$i(t,k) = \sigma\,(W \quad x'(t,k) + W \quad h(t-1,k) + W \quad c(t-1,k) + b \quad)$$

$$c(t,k) = f(t,k)c(t-1,k) + i(t,k)\tanh\,(W \quad x'(t,k) + W \quad h(t-1,k) + b \quad)$$

$$o(t,k) = \sigma\,(W \quad x'(t,k) + W \quad h(t-1,k) + W \quad c(t,k) + b \quad)$$

$$h(t,k) = o(t,k)\tanh\,(c(t,k))$$

$$y'(t,k) = W \quad h(t,k) + b$$

3.2.1. Deep-LSTM Reader

1. 알파이자 오메가

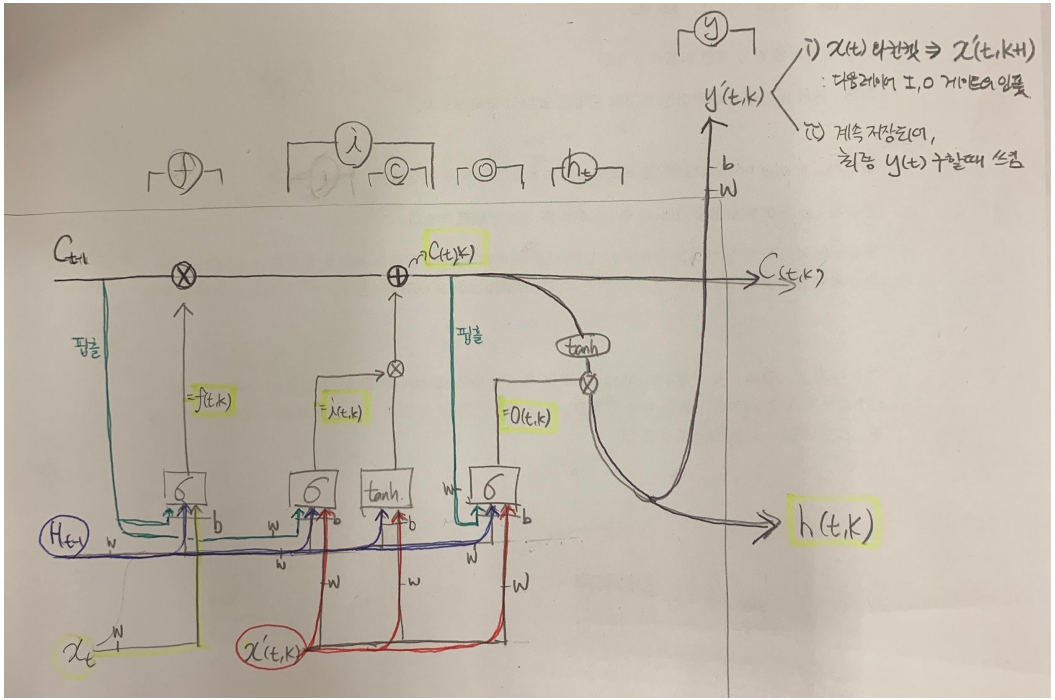
$$x'(t,k) = x(t)||y'(t,k-1), \qquad y(t) = y'(t,1)||\dots||y'(t,K)$$

3.2.1. Deep-LSTM Reader

2.포겟게이트

ㄱ.초록선=핍홀=C_{t-1}

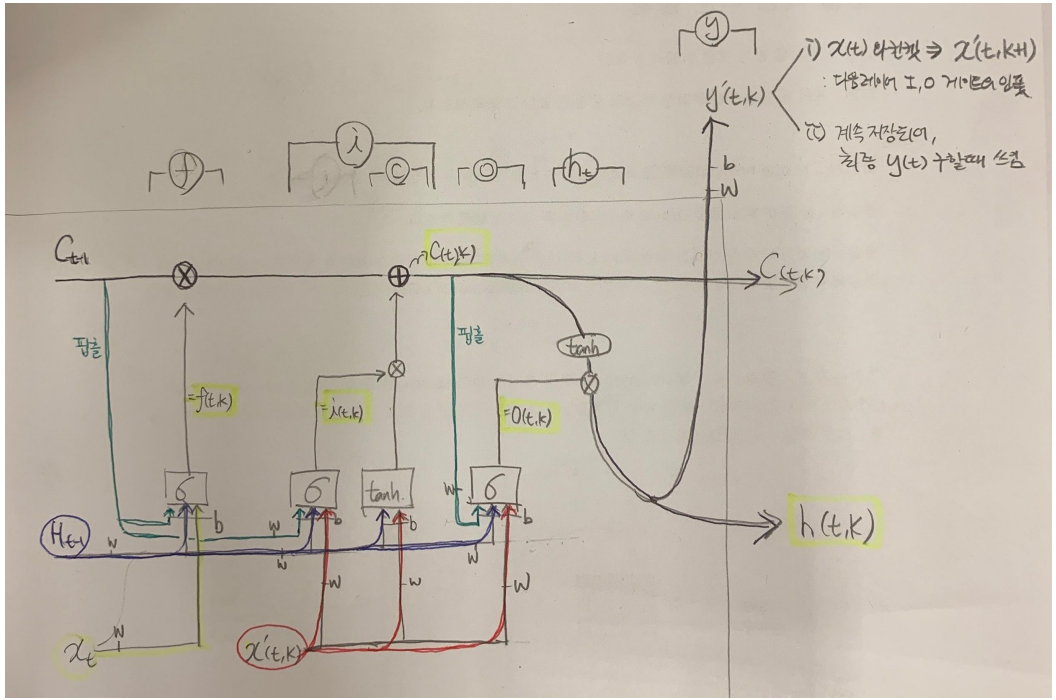
ㄴ. f게이트만 x` (전 레이어의 y`와 연관)
대신 x (원래인풋자체)를 받음



$$f(t,k) = \sigma\left(W \quad x(t) + W \quad h(t-1,k) + W \quad c(t-1,k) + b \quad\right)$$

3.2.1. Deep-LSTM Reader

3. 인풋게이트



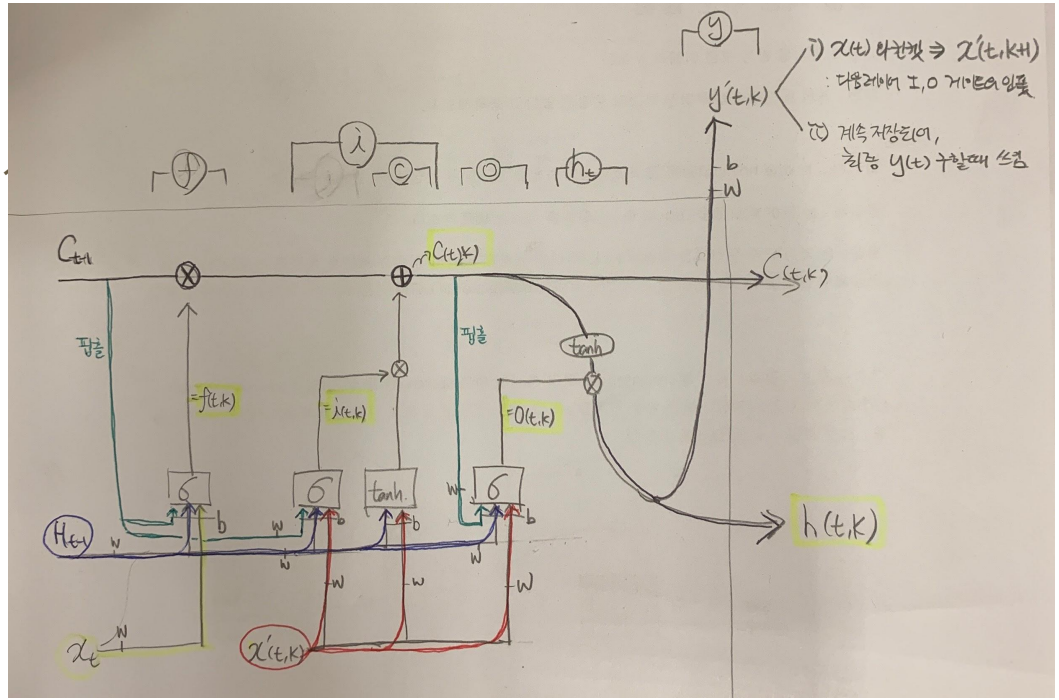
$$i(t,k) = \sigma\,(W \quad x'(t,k) + W \quad h(t-1,k) + W \quad c(t-1,k) + b \quad)$$

3.2.1. Deep-LSTM Reader

4.Ct

(히든스테이트

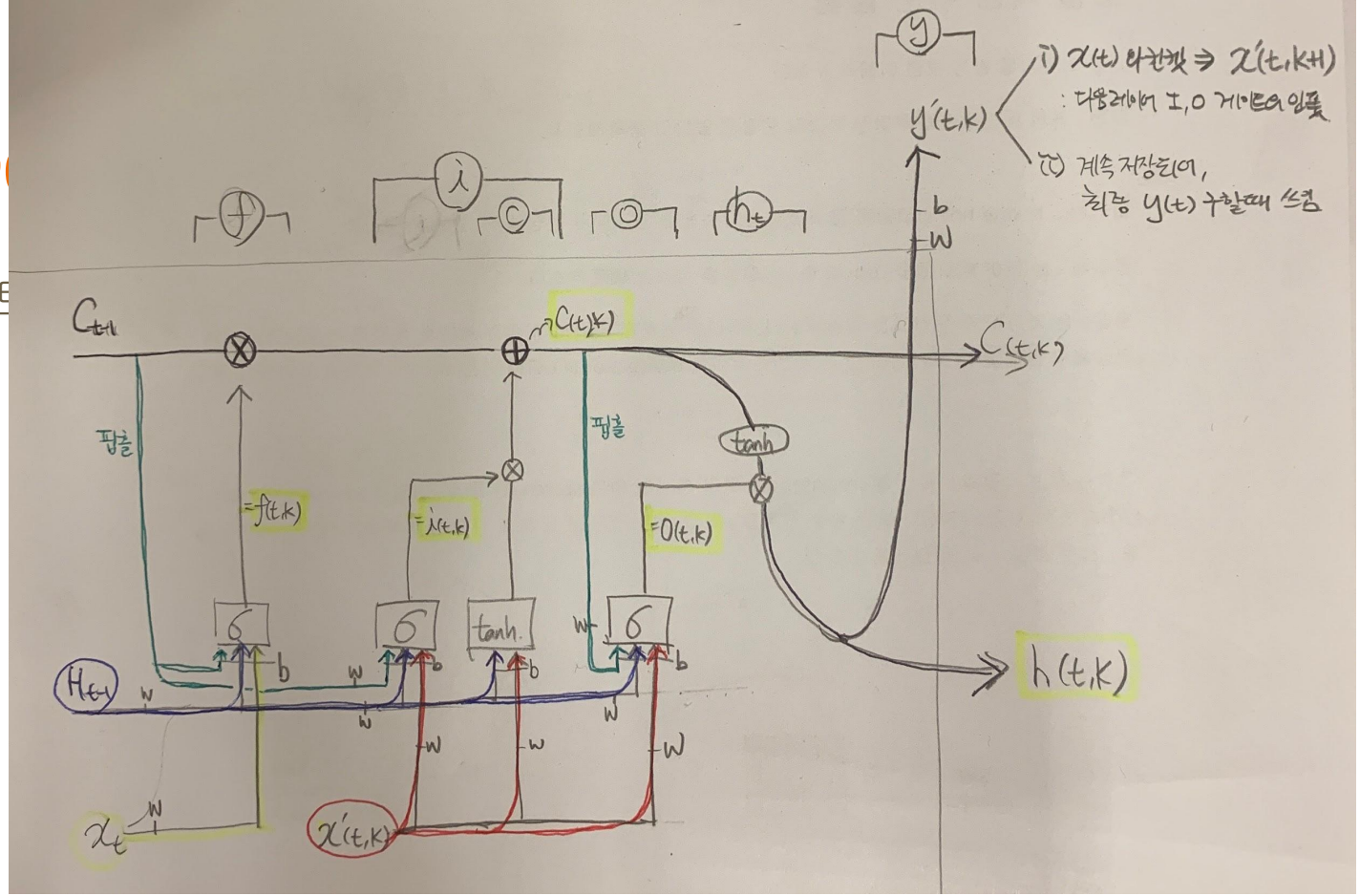
장기기억)



$$c(t,k) = f(t,k)c(t-1,k) + i(t,k)\tanh(W \quad x'(t,k) + W \quad h(t-1,k) + b \quad)$$

3.2.1. De

5. 아웃풋게이트



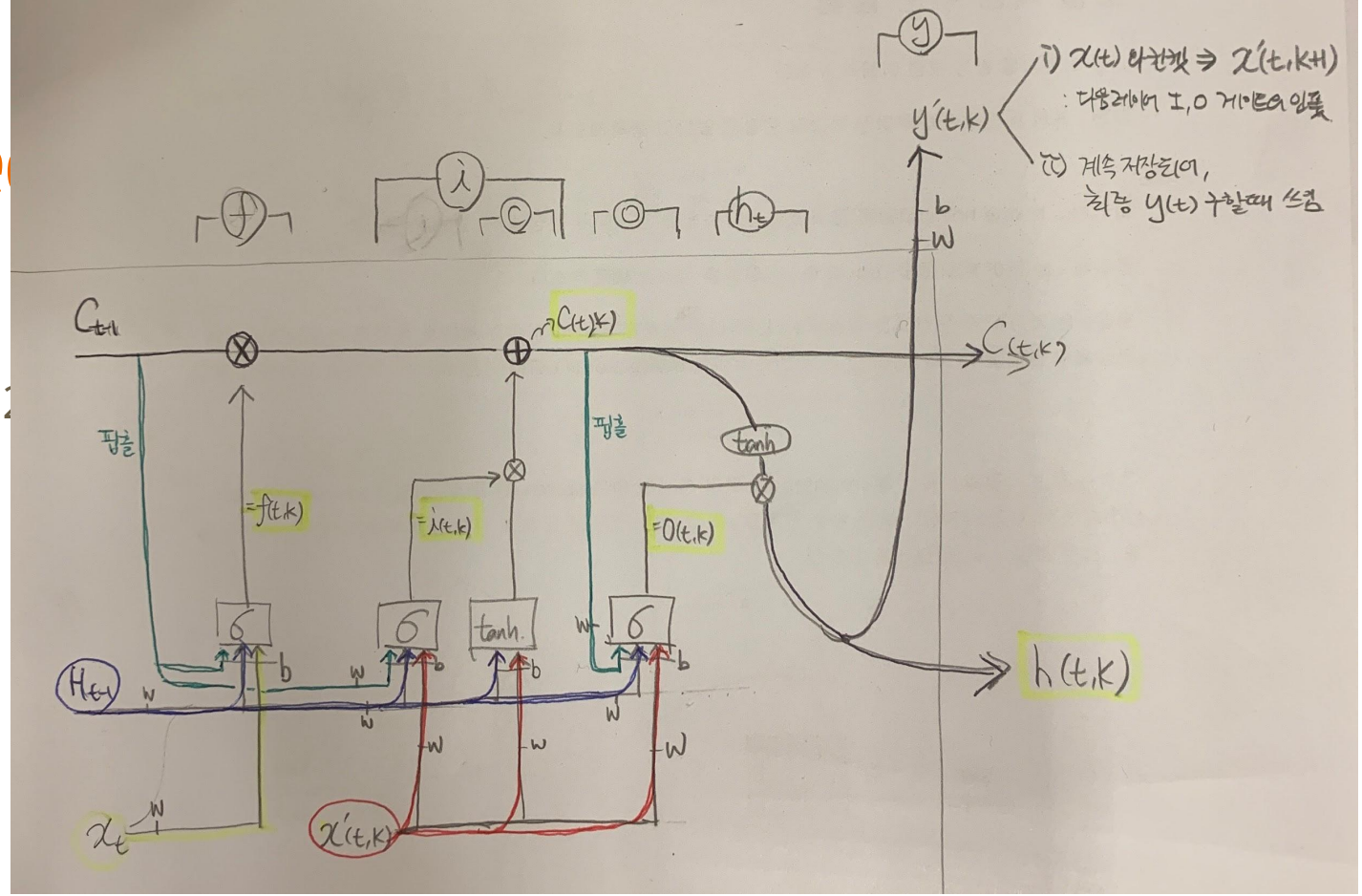
$$o(t,k) = \sigma\left(W \quad x'(t,k) + W \quad h(t-1,k) + W \quad c(t,k) + b \quad\right)$$

3.2.1. De

6. Ht

(히든스테이트

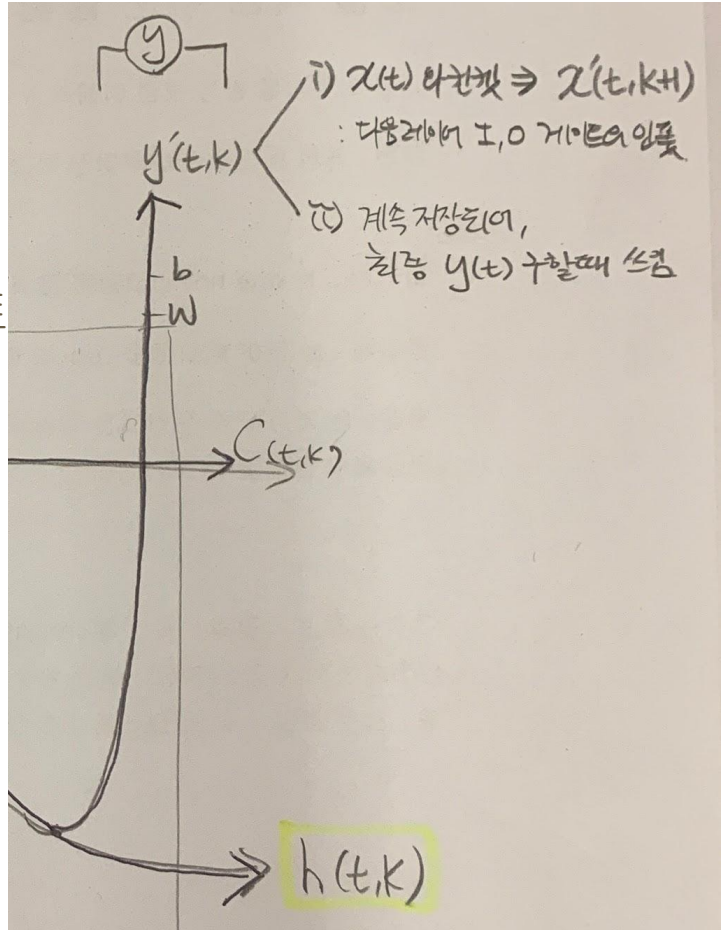
단기기억)



$$h(t,k) = o(t,k)\tanh\left(c(t,k)\right)$$

3.2.1. Deep-LSTM Reader

7. y` 와 그를 통한 다음레이어(k+1)의 x` 업데이트



$$y'(t,k) = W \quad h(t,k) + b$$

$$x'(t,k) = x(t)||y'(t,k-1), \qquad y(t) = y'(t,1)||\ldots||y'(t,K)$$

3.2.1. Deep-LSTM Reader

래츠고님 질문 : 왜 f게이트"만" 빼고 I,O게이트에"만" X || Y' 인풋으로 넘기나?

- (내생각)
  - f게이트는 원래인풋(x)만 받고,
  - i랑o게이트는 1.원래인풋과 2. 이전레이어의 연산결과 (y')가 어느정도 섞인 (x') 를 받는데,
    - 이**전** 레이어의 결과는 "그 레이어"의 장기(Ct)와 단기(Ht) 기억을 반영하여 y'를 만들었고,
    - 이**번** 레이어는 y'가 "단순컨켓"된 x' , 즉 전레이어의 영향을 받은 변형된 인풋을 받는다.
      - ㄴ> 이거 너무 원시적이지않음?
    - 따라서,
      - 새 레이어의 i/o게이트는,
      - "모든 이전 레이어들"의 비선형 연산결과가 어느정도는 섞여들어간 지난레이어의 y'에 영향을 받은 x'를,
      - 이번 i/o 게이트내의 학습파라미터들에 반영한다.
  - 결과적으로, 이번 레이어는, 앞선 레이어들의 연산결과를 "인풋"과 "아웃풋"게이트에만 반영하는데,
  - 그럼 왜? f는 제외한거지?
    - 1) 그냥 deep LSTM의 스킵커넥션 디자인이 그래서
    - 2) 다른 이유가 있어서.
      - 예를들어, 레이어가 쌓일때마다 - "모든 이전 레이어들"의 연산이 어느정도 반영된 인풋을 받으므로, 원래 딥 뉴럴넷이 가지는 비선형사상 중첩효과를 가질수있어서?
      - 논문 impatient reader 부분에 ; 반복적으로 문서의 정보를 누적 할 수있게 해주는 어텐션 메커니즘

3.2.2 Attentive Reader

아키텍쳐 : separate bidirectional **single** layer LSTMs

- “Single layer”에서 느껴지는 패기
-

- 어텐티브 리더는 QA용 **메모리 네트워크**의 일반화된 적용으로 볼수있다. 이 모델의 어텐션 메커니즘은 문장단위로 적용되는데, 각 문장은 (토큰)임베딩들의 bag으로써 표상화된다. 어텐티브리더는 , 매우 세분화된 (인풋 문서의 과거/미래의 문맥을 기반으로 임베딩된) 토큰 레벨에서의 어텐션 메커니즘을 사용.
  - 내 궁금 : 이게 왜.. 차라리 못참는독자가 메모리네트워크랑 비슷한거아냐?? 그럼 임페이션트 = 종단메모리네트웤이랑 비슷한거고, (여러번읽고, 에피소딕 메모리 갱신?), 어텐티브 = 그냥 메모리네트워크랑 비슷한거고?
    i. 메모리넷 : 스토리 선택모듈 / 답변선택모듈
- 아니야아니야. 메모리넽웤과 비슷한거맞음 :
  - 원문에 ; the attentive reader is able to focus on the passages of a context doc that are most likely to inform the answer to the query. 라는게, 메모리넷웤의 스토리 선택모듈과 ㅂ슷한거 같엉 .
  - 근데 Deep LSTM은 그런 기능이 아예없지,
    i. 다만 전 레이어의 연산결과 y’를 이번 레이어의 i,o 게이트에 “만" 반영할 뿐인거지,
    ii. 즉 여러번 보는것 비슷(에피소딕 메모리) 한데, 어텐션 기능은 없는것같음.

3.2.2 Attentive Reader

Attention 연산 기본포맷 :

A = softmax (tanh (W*input)) ; A는 인풋 각각의 (y레이블에대한) “중요정도"를 백프롭으로 학습한 후 확률값이 된, 행벡터.

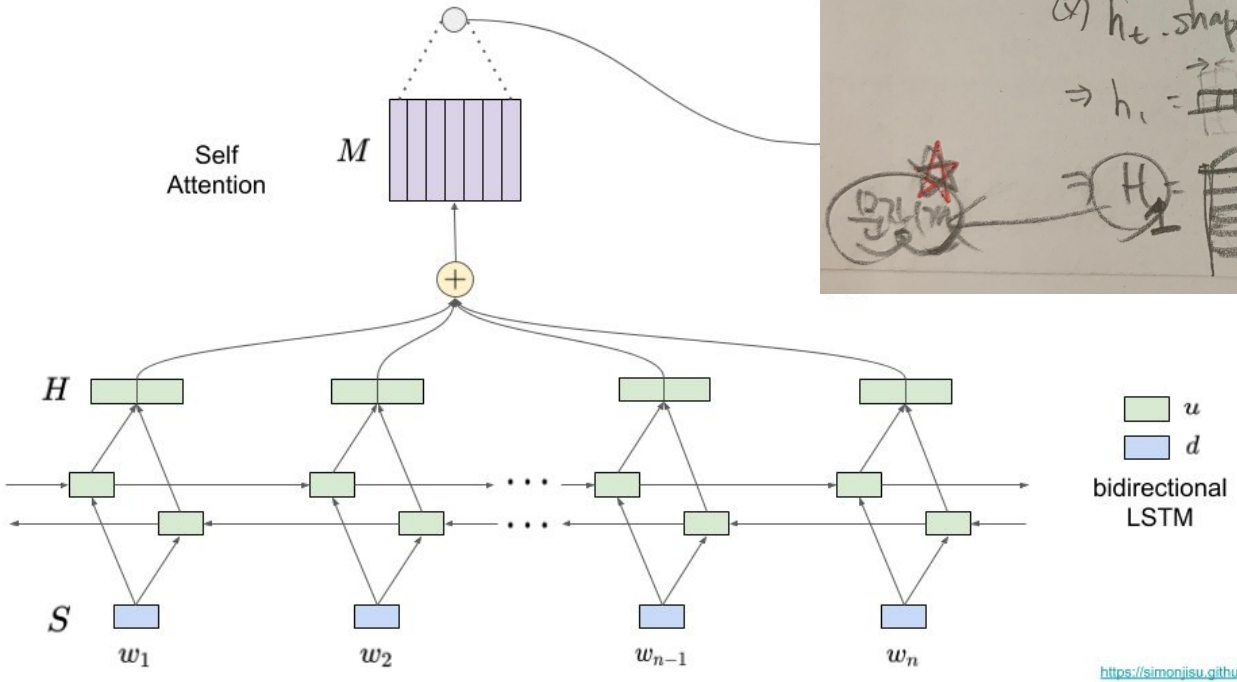
W -> 비선형사상 -> 소맥 연산

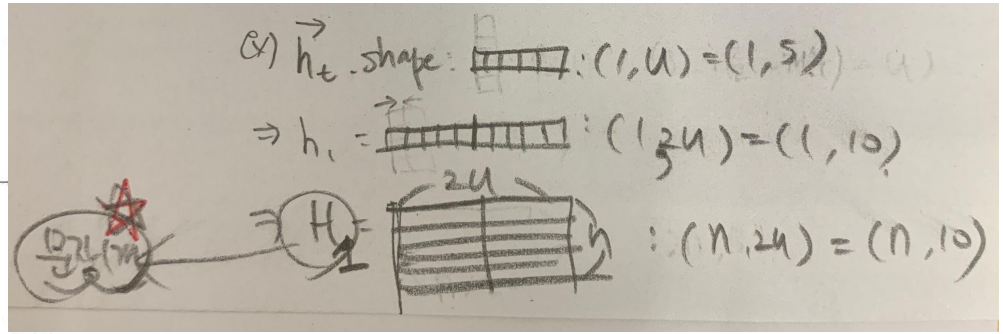
- = A의 모든 내부 엘레멘트의 합 = 1 (확률) & A 는 보통 행벡터.
- 예) 문장 토큰 5개 -> a.shape = (1,5) 의 엘레멘트 값이 [0.1, 0.2, 0.1, 0.1, 0.5] 라면 ; 5번째 토큰이 젤 중요

A * 토큰 표상 = “가중치가 부여된” 새로운 토큰&문장 representation

But 만약 문장 내 중요한부분이 여러개있다면? 여러번 (hops)해야한다. 이를 query token 개수만큼 하는게 딥의 impatient reader 이다

3.2.2 Attentive Reader



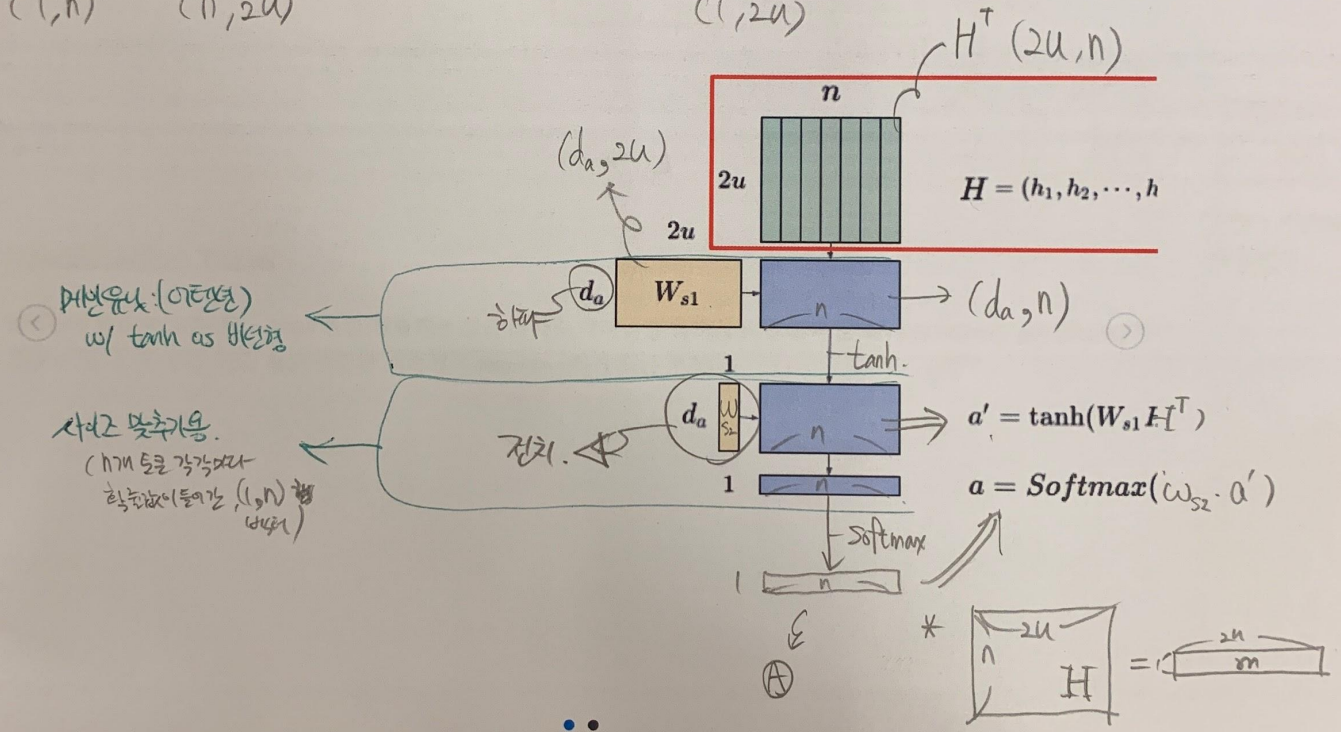


https://simonjisu.github.io

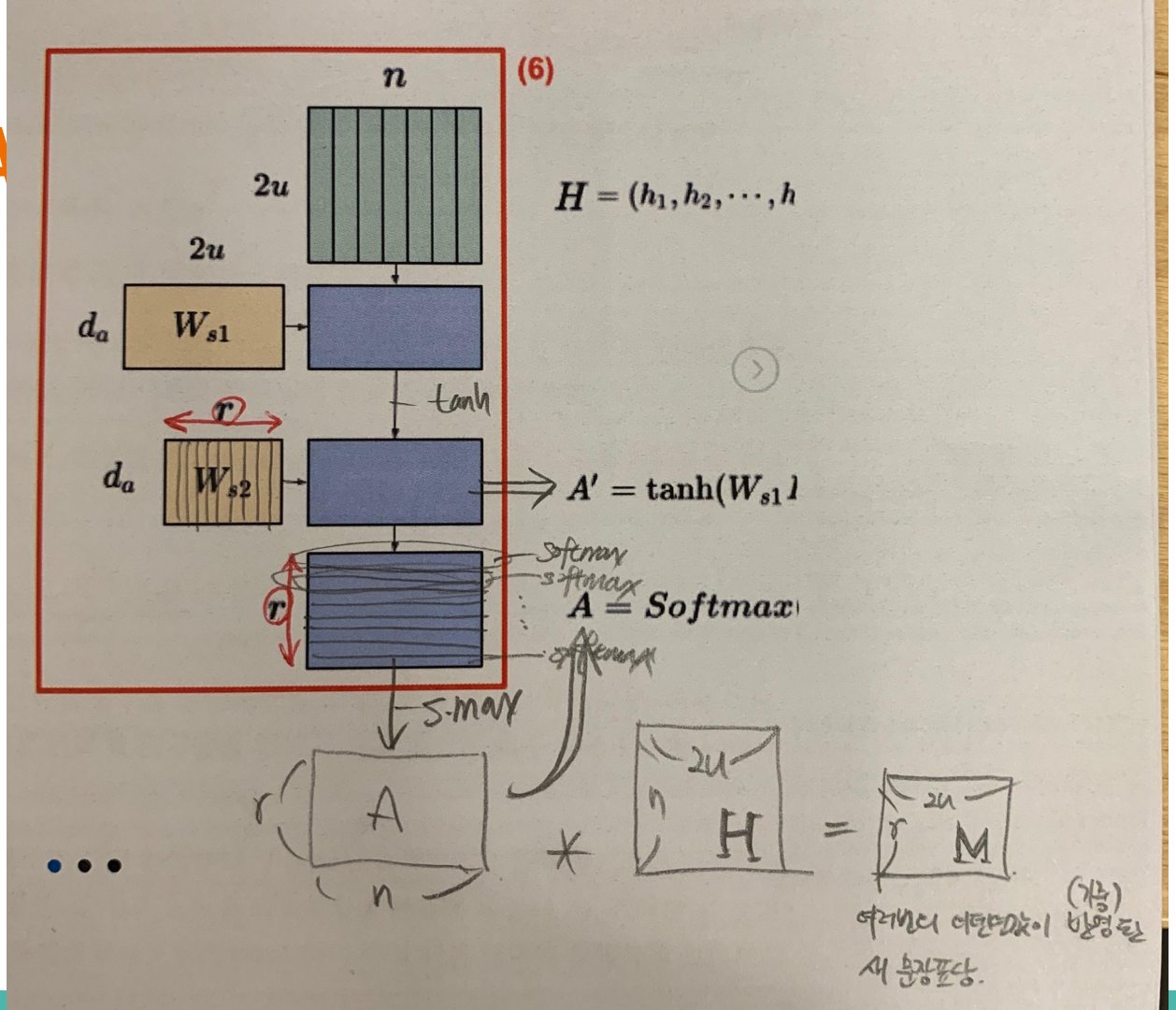
3.2.2

각 단어를 input으로 받은 hidden 상태의 노드들은 단어를 통과해서 각 단어의 숨겨진 특성을 대표하고 있다. 학습 시 Task 에 따라 다르겠지만, 분류라고 가정한다면 분류에 도움이 되는 히든 상태는 높은 값을 가지게 될 것이며, 이를 어떤 선형 변환 과정을 거쳐 softmax 취한다는 것은 한 문장에서 분류에 도움이 된 근거 단어 혹은 중요 단어의 확률을 구한다는 것이 된다. (그래서 attention 이라고 하는 것 같다.) 따라서 이는 한 문장에서 **의미적인(semantic)** 부분을 나타내고 있다고 할 수 있다.

이 확률 a 를 기존의 hidden 상태와 곱해서 의미부분을 조금더 강조하게 되는 벡터 m 을 구했다고 보면 된다.



3.2.2 A


n
(6)
2u
2u
$H = (h_1, h_2, \cdots, h$
d_a
W_{s1}
tanh
r
d_a
W_{s2}
$A' = \tanh(W_{s1}$
Softmax
Softmax
$A = Softmax$
Softmax
r
S.max
A
r
n
2u
n
H
=
2u
r
M
여러번의 여러번째의 비슷한 (가능)
새 문장표상.

3.2.3. Impatient Reader

- 주의깊은 리더는 , 질문에 답할수있는 정보를 가장 잘주는 구절을 문서에서 찾아낼 수 있다.
- 이를 좀 더 발전시켜서, 쿼리의 토큰들이 각각 읽어질때마다, 모델이 문서를 다시 읽도록 변형. (~~ 에피소딕 메모리)
- 그 결과 모델이 각 쿼리 토큰을 보면서 반복적으로 문서의 정보를 누적 할 수있게 해주는 주의 메커니즘이 생겨, 궁극적으로 대답 예측을 위한 최종 문서-쿼리 공동표상이 출력됩니다.
- 비교 ;
  a. (약간유사) 종단간 MN : 질문과 관련된 문장이 여러개일경우에 **만** 메모리에 여러번 접근 & 연산
  b. (많이 유사) 다이나믹 MN : 설정한 에피소드 개수만큼 문서 리딩
  c. Impatient reader : 쿼리의 토큰 개수만큼 문서 리딩

3.2.3. Impatient Reader

죄송합니다. 솔직히 이 수식은 잘 모르겠습니다.

The Impatient Reader The Attentive Reader is able to focus on the passages of a context document that are most likely to inform the answer to the query. We can go further by equipping the model with the ability to reread from the document as each query token is read. At each token i of the query q the model computes a document representation vector $r(i)$ using the bidirectional embedding $y_q(i) = \overrightarrow{y_q}(i) \mid\mid \overleftarrow{y_q}(i)$:

$$m(i,t) = \tanh\left(W_{dm}y_d(t) + W_{rm}r(i-1) + W_{qm}y_q(i)\right), \quad 1 \le i \le |q|,$$
$$s(i,t) \propto \exp\left(\mathrm{w}_{ms}^{\intercal} m(i,t)\right),$$
$$r(0) = \mathbf{r_0}, \quad r(i) = y_d^{\intercal} s(i) + \tanh\left(W_{rr}r(i-1)\right) \quad 1 \le i \le |q|.$$

The result is an attention mechanism that allows the model to recurrently accumulate information from the document as it sees each query token, ultimately outputting a final joint document query representation for the answer prediction,

$$g^{\mathrm{IR}}(d,q) = \tanh\left(W_{rg}r(|q|) + W_{qg}u\right).$$

3.2.3. Impatient Reader

라츠고 님 질문 : 왜 소맥 안하지? 원래 softmax(m) = s 여야하지않음?

The Impatient Reader The Attentive Reader is able to focus on the passages of a context document that are most likely to inform the answer to the query. We can go further by equipping the model with the ability to reread from the document as each query token is read. At each token i of the query q the model computes a document representation vector $r(i)$ using the bidirectional embedding $y_q(i) = \overrightarrow{y_q}(i) \mid\mid \overleftarrow{y_q}(i)$:

$$m(i,t) = \tanh\left(W_{dm}y_d(t) + W_{rm}r(i-1) + W_{qm}y_q(i)\right), \quad 1 \le i \le |q|,$$
$$s(i,t) \propto \exp\left(\mathrm{w}_{ms}^{\intercal} m(i,t)\right),$$
$$r(0) = \mathbf{r_0}, \quad r(i) = y_d^{\intercal} s(i) + \tanh\left(W_{rr}r(i-1)\right) \quad 1 \le i \le |q|.$$

The result is an attention mechanism that allows the model to recurrently accumulate information from the document as it sees each query token, ultimately outputting a final joint document query representation for the answer prediction,

$$g^{\mathrm{IR}}(d,q) = \tanh\left(W_{rg}r(|q|) + W_{qg}u\right).$$

4 Empirical Evaluation

- 이 논문의 실험적 파트는 3폴드 목적
  1. 기계리딩 태스크에 좀 더 어려운 테스트 만들기
     i. = 좀 더 풀기 “어려운" 문제집 생성
        1. 3년간 공부 하나도안해도 수능보면 200점은 나옴 => “world knowledge”
        2. 상식이고 뭐고 “**수능공부**” 안하면 0점맞는게 좋은 수능 아니냐?
  2. 뉴럴모델과 parse-based? models?를 비교할것이다.
     i. → NN이 낫다 (고봐도 무방)
  3. LSTM과 Attention 각각이 퍼포먼스에 얼마나 영향을 끼치는지 분석하는것
     i. → 어텐션 짱짱맨

재미로보는 구글 vs 파파고(3)

- ?The entity anonymisation and permutation aspect of the task presented here may end up levelling the playing field in that regard, favouring models capable of dealing with syntax rather than just semantics.?
-
- 구글 : 여기에 제시된 과제의 개체 익명화 및 순열 측면은 그 의미에서 경기장을 평평하게 만들 수 있으며, 의미론보다는 문법을 다룰 수있는 모델을 선호합니다.
-
- 파파고: 여기서 제시된 과제의 익명화와 허용 측면은 의미론뿐만 아니라 구문을 다룰 수 있는 모형을 선호하면서 그러한 측면에서 경쟁 분야의 수준을 조정하는 결과를 가져올 수 있습니다.

4.1, 4.2 솔직히 뭔말인지 모르겠어요 ㅠ

4.1 Frame-semantic benchmark

4.2 Word distance benchmark

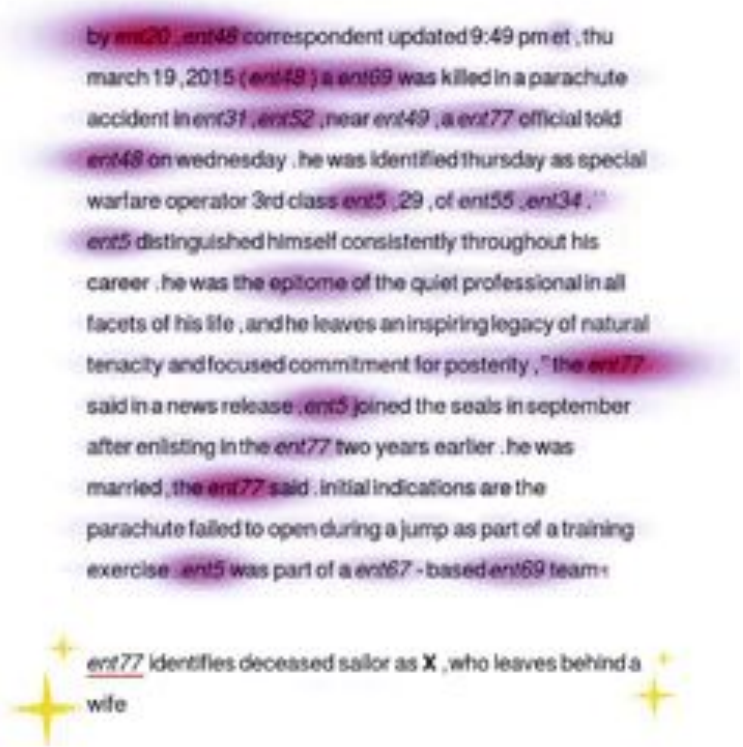
4.3 NN models

결론 = 어텐션 짱짱맨

- 어텐션은 싱글레이어 쓰고도 , 멀티레이어 LSTM 능가

Visualization : impatient model

질문의 토큰을 읽을때마다어디에 어텐션을 주는



쿼리의 “단어"가 변함에 따라, 어텐션 토큰 변화

5. Conclusion

앞으로 나아가야 할길 :

- world knowledge 와 다중문서질의를 통합하기위해서는, 어텐션/임베딩 메커니즘의 발전을 필요로 할것입니다.
- 왜냐하면 쿼리의 복잡성은 데이터셋 사이즈에 선형비례하지않기때문입니다.
  - 예) 머신이 얼마나 알든말든 , 유저는 자기맘대로 얼마든 복잡한 질문을 날릴수 있음
- 여전히 우리 모델이 풀지못하는, 복잡한 추론과 장거리 레퍼런스가 필요한 쿼리들이 많이있다.