question answering

CS585, Fall 2019

Deep Learning for Natural Language Processing

Mohit Iyyer

College of Information and Computer Sciences

University of Massachusetts Amherst

some slides from Jordan Boyd-Graber & Kalpesh Krishna

questions from last time

- final projects? how much work do we have to put in? implement everything from scratch?

- HW3 due Thursday

- If you want CSCF to print your poster, submit it by this Friday at noon

Song Genre Classification

too much text!

Introduction

Our project objective was to run various natural language processing classification algorithms on a dataset of songs to compare the effectiveness of these algorithms in identifying the genre of the songs.
We used a bag of words representation of the song lyrics linked to ground truth genre tags to train the algorithms and then predict genres for new sets of lyrics.

Dataset information

Our dataset contains 13 genres with a distribution of:

Pop_Rock 75.15%
Reggae 0.70%
Country 4.00%
Jazz 0.50%
Vocal 1.06%
New Age 0.16%
Latin 4.30%
Rap 4.06%
RnB 3.93%
International 1.78%
Blues 0.57%
Electronic 2.78%
Folk 1.00%

- dataset is a BOW representation of the stemmed lyrics
- Derived from Million Songs Dataset
- Split 90-10 training vs test
- 114,643 songs in the dataset

Approach

We were unable to find a dataset that linked lyrics directly to genre, so we first had to compile information from multiple datasets into one that we could use. The musiXmatch dataset maps songs to lyrics while the MSD Allmusic Top Genre Dataset maps songs to genre, creating the perfect combination for what our project needed. Once we had our data, we began implementing different natural language processing algorithms using python's scikit-learn library. After training these algorithms on a large percentage of our dataset and testing their ability to correctly classify the remaining portion, we were able to identify which type of algorithm generally

Results

- **Decision Tree Algorithm**: 70.06% accuracy
- **Multi-Layer Perception**: 76.45% accuracy
- **Stochastic Gradient Descent (SGD)**: 76.16% accuracy
- **Support Vector Machine Classifier (SVM)**: 75.22% accuracy
- **Voting Classifier**: 78.51% accuracy

The Voting Classifier used the other algorithms and implemented a voting system such that each classifier had a say in the genre assigned to a given example. This turned out to get a small boost in accuracy over the other classifiers as it could weed out any outliers when one of the algorithms predicted the wrong result.

The Multi-Layer Perceptron and SGD classifiers performed a bit better than the others

Conclusions

- We were unable to use many of the more "advanced" algorithms on our dataset due to its limitations as a pre-stemmed/lemmatized BOW representation of the lyrics.
- Given more time/resources it probably be possible to compile a "better" dataset which we could run algorithms that would obtain higher accuracy.

References

https://labrosa.ee.columbia.edu/millionsong/

http://web.stanford.edu/class/cs224n/reports/2728368.pdf

https://nlp.stanford.edu/courses/cs224n/2006/fp/sadovsky-x1n9-1-224n_final_report.pdf

too much text!

Twitter Sentiment Classification and Analysis

Purpose

The purpose of this project is to predict the sentiment of a tweet based on a 5-point scale (from very negative to very positive) and compare the sentiment of the topic of a tweet among various demographics through graphs.

We have previously classified text sentiment based on a two-point scale (negative versus positive) in class, so this project is meant to push the boundaries. Because the source of the tweet data also provides user demographic data, it seemed interesting to visually analyze sentiment trends based on a user's location.

Method & Results

Of the many ways to classify sentiment, the first attempted for this project was the Naïve Bayes, bag-of-words method, where the tweets are tokenized and evaluated based on each individual token. The classifier is trained on the tokens stored in each sentiment dictionary (one for each rating on the scale) based on the provided sentiment of the tweets in the training data.

I additionally attempted to include an external dictionary with generally known words and their sentiment weights to add to the weights calculated during the classifier training. When comparing the two implementations, the external dictionary proved to hurt rather than help the classification accuracy.

While the classification accuracy remained above 50% on all data sets, this method proved inefficient compared to others learned in class.

Future Work

Because the bag-of-words method was found to be inefficient, I am currently working on implementing a classification perceptron method to replace it, since it proved to have a much higher accuracy when compared to the Naïve Bayes method.

In addition, graphs displaying the sentiment among users from different location have yet to be created. There will be two types of graphs: the first will show the sentiment across a single group on a single topic, and the second will compare the general sentiment (if there is a clear one) of two different groups on a single topic.

Data and Tools

The SemEval-2017 Task 4 Data and Tools page provided all of the needed materials for obtaining the data for this project. This data included training, development, and testing sets for tweets written in English, as well as information about the users who wrote the tweets. For reading and parsing reasons, the data needed to be cleaned using a script.

Tools used:

- Python 2.7.13
- Natural Language Toolkit (NLTK)
- Matplotlib

Graphs

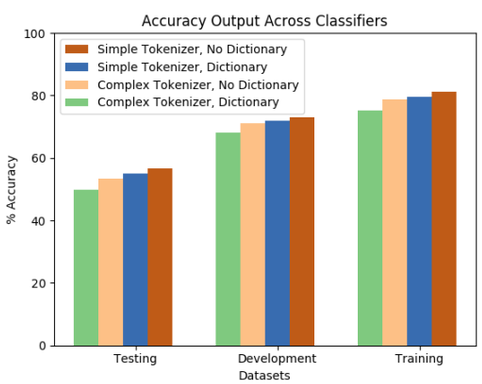


Fig 1: Multi-bar chart to compare accuracy outputs across classifier implementations on different datasets

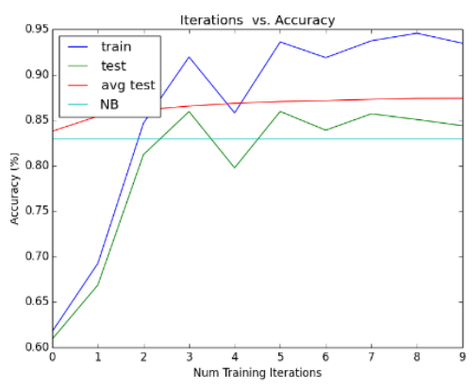


Fig 2: Line chart taken from the solutions of a previous homework displays the anticipated accuracy of the perceptron implementation

References

- Farra, N., Nakov, P., & Rosenthal, S. (2016). *SemEval-2017 Task 4: Sentiment Analysis in Twitter*, SIGLEX. Retrieved from alt.qcri.org/semeval2017/task4/
- Taboada, M., Brooke, J., Voll, K., Anthony, C., & Grieve, J. (2009). SO-CAL (Version 1.11). github.com/DrOttensooser/BiblicalNLPworks/tree/master/SkyDrive/NLP/CommonWorks/Data/Opion-Lexicon-English/SO-CAL

Price Prediction of Alternative Cryptocurrencies using Telegram Group Chats

could have less text, overall not bad!

Overview

This project uses existing sentiment analysis and machine learning techniques to anticipate price movements of alternative cryptocurrencies using popular Telegram chat groups. Telegram is a popular chat application that has been adopted by cryptocurrency communities for price speculation, and as an interface between project teams and the community. Since Bitcoin is the de facto bridge between fiat and all other cryptocurrencies, backtesting against the market will be evaluated according to maximization of a simulated Bitcoin account.

Datasets

Coin	Ticker	Telegram Chat	Members	Msg / Hour
Litecoin	LTC	Litecoin LTC	8535	36.5
XEM	XEM	NEMberia 2.0	1768	17.7
Ethereum	ETH	EthTrader	5046	14.3

Sentiment Lexicon

A random subset of messages in Litecoin LTC was manually annotated as displaying strong positive or negative indications of sentiment or outlook regarding price. Using the results of annotations, a custom lexicon was developed by hand using the keywords found with sentiment weights. This lexicon used generic keywords allowing it to be reused for other cryptocurrencies. Cryptocurrency slang (e.g. 'mooning'), trading terminology ('long', 'short'), and common slang ('rekt') were incorporated into the lexicon, in addition to words in the existing VADER lexicon.

Relevance Annotations

Messages were manually annotated according to perceived relevance to the coin or its market behavior.

Chat	Annotations	Relevant	Irrelevant
Litecoin LTC	3207	1248	1959
NEMberia 2.0	3295	875	2422
EthTrader	2682	474	2208

Relevance Classifier (Neural Net)

A multi-level perceptron classifier with a single hidden layer of size 50 was trained on single word ngrams of the training set's annotated data with 10,000 iterations. Train/Test split was done on 07/01/2017.

	Train Size	Dev Size	Precision	Recall	F1
LTC	2511	694	.70	.69	.68
XEM	2425	721	.78	.80	.78
ETH	1860	816	.80	.81	.81

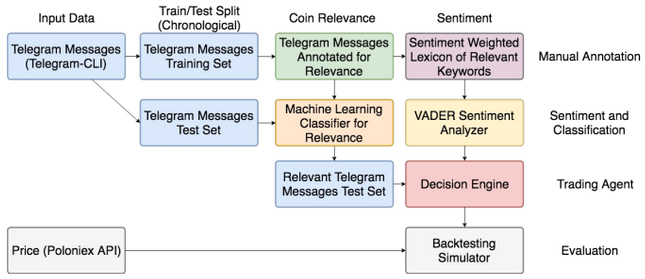
VADER Sentiment Analysis and Granger Causality

Granger causality was calculated based on VADER sentiment and price, using custom and stock lexicons. This established correlation between the price and sentiment time series expressed with both lexicons.

	Max Time Lag w/ p value > .05	
	Stock Lexicon	Custom Lexicon
LTC	>15h	>15h
XEM	8h	9.5h
ETH	8.5h	8h

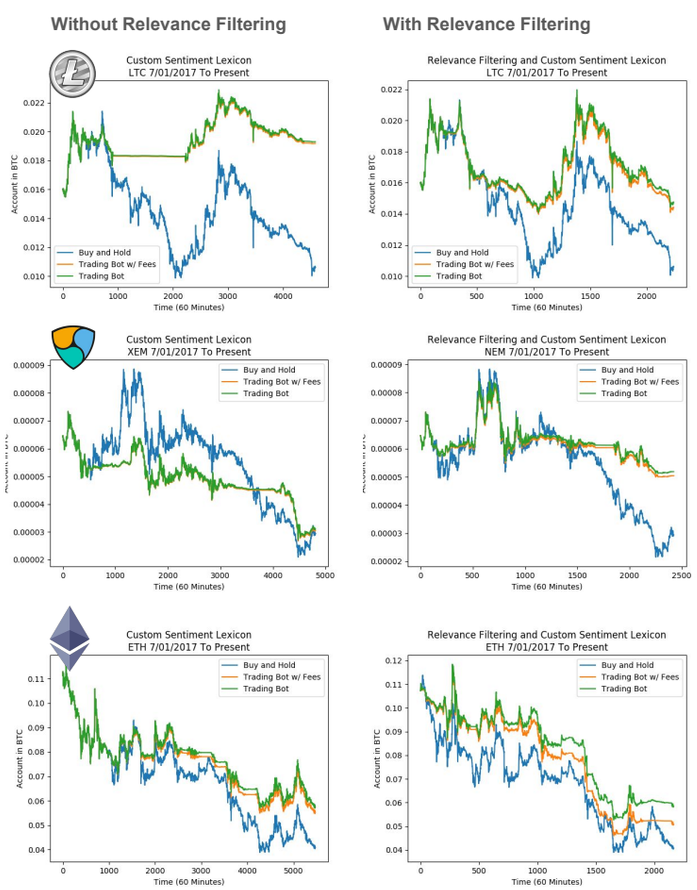
Trading Algorithm

Sentiment was calculated for each 60 minute group of messages, and a exponential weighted moving average (EWMA) of sentiment, and deviation is maintained. When sentiment rises or drops above the EWMA of sentiment past a deviation threshold, a percentage of the altcoin account proportional to the difference between sentiment and sentiment EWMA is transferred to the altcoin, or Bitcoin account, respectively.



Evaluation

The trading agent with relevance filtering outperformed buy and hold both with, and without a standard .25% transaction fee for each order made.



pretty good!

Aspect Extraction using Dependency Parsing and Semantic Clustering

Problem Description

" it gives **great** **pictures**,
the **controls** are **easy to use**,
the **battery** **lasts forever** on one single charge,
but the **software** is **not user-friendly** at all! "

Pictures	great
Controls	easy to use,
Battery	lasts forever
Software	not user-friendly

Pictures
Controls
Battery
Software

Procedural Steps

Dependency Parse

Rule-Based Extraction

VB ← advmod ← RB
NN ← acomp ← JJ
NN ← amod ← JJ
...
NN ← nsubj ← VB → dobj → NN
RB ← advmod ← VB → nsubj → NN

Heuristics

- VB ← neg ← RB
- Delete Stop Words
- Ignore Low Frequency

Opinion Seed Lexicon

Domain Seed

Camera
Tec h
Display
Electronic

Aspect Filtering using Word Embeddings

→ Photos

Crisp
Clear

Results

	Aspect Precision	Aspect Recall	Opinion Precision
DVD Player	0.316	0.201	0.492
Camera-1	0.347	0.487	0.596
Camera-2	0.516	0.534	0.341
MP3 Player	0.360	0.411	0.571
Cell Phone	0.545	0.525	0.478
OVERALL	**0.385**	**0.384**	**0.504**

Further Work

- More Heuristics
- Recursive Seed Expansion
- Better Semantic Clustering

- ***Hu and Liu. 2004. Mining and summarizing customer reviews***, *10th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*
- ***Qiu, Liu, Bu, and Chen. 2011. Opinion word expansion and target extraction through double propagation.*** *Computational Linguistics*

good!

Wordify: A Reverse Dictionary for Everyone

COMPSCI 585
Introduction to Natural Language Processing

Task

Ever had a word at the tip of your tongue and still be unable to speak or write it?

Using a Reverse Dictionary, you can turn your thoughts into words!

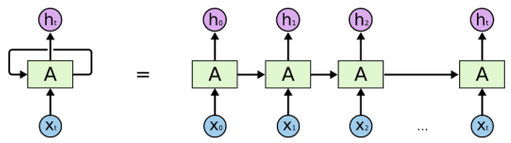
Aim: Develop a reverse dictionary by learning to map the definitions in a dictionary to the word embeddings of the words that they define.

A native of a cold country - **eskimo**
A way of moving through the air - **glide**

Approach

Step 1 Learn word embeddings using Word2Vec

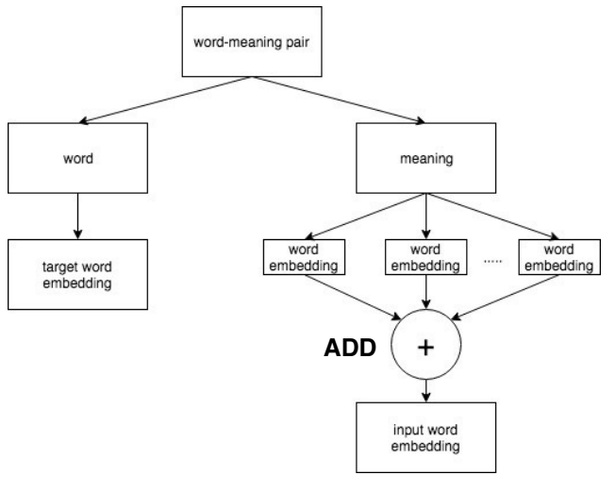
Step 2 Train a RNN to map the sentence or phrase to the word embedding of the word that it defines



Step 3 Map the input phrase to a point in the embedding space and return the words closest to that point

Progress so far….

- **Collected data from WordNet**
- **Processed and stored the data**
- **Used gensim to create word embeddings**
- **Implemented two baseline algorithms**



Preliminary Results

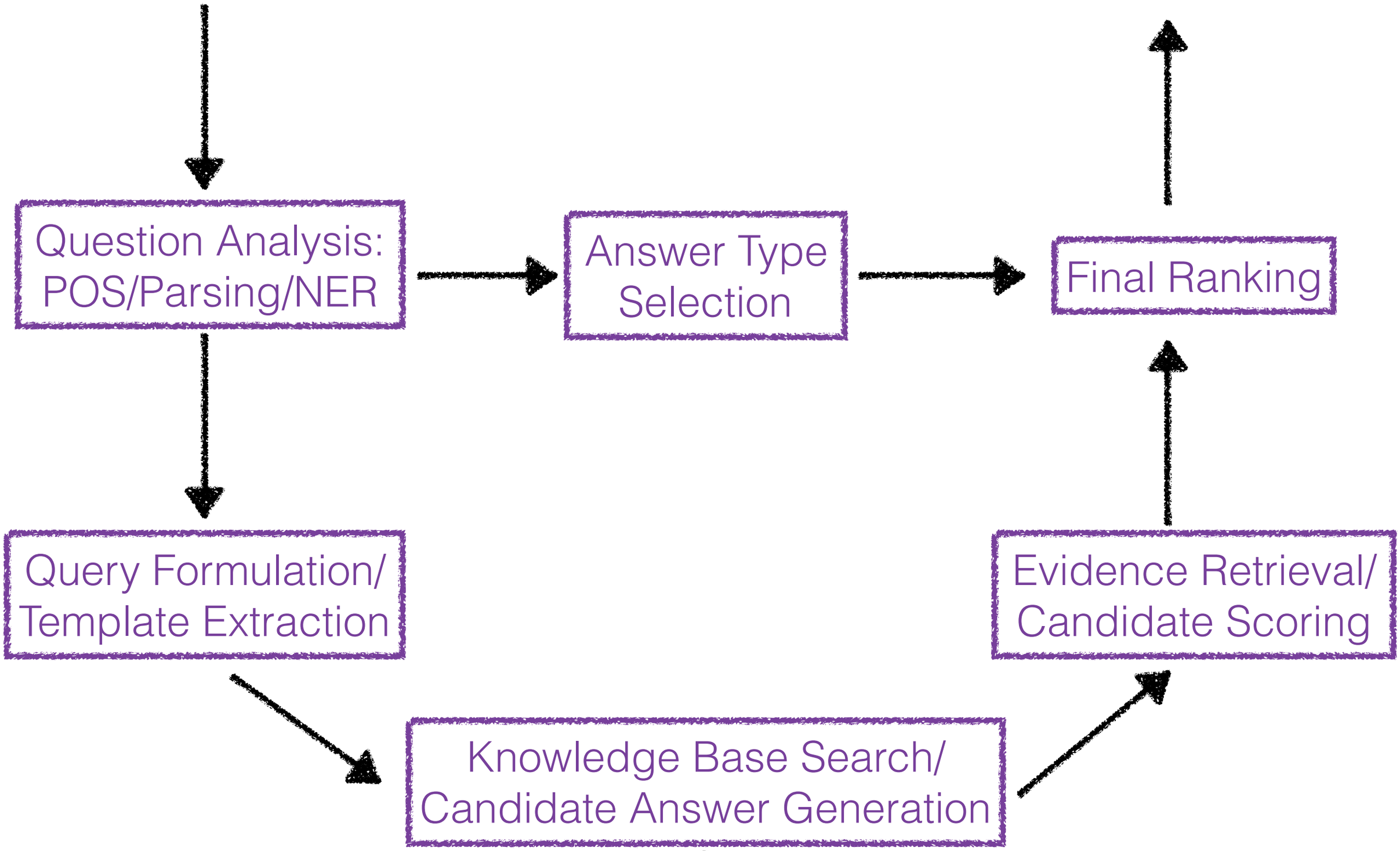
Baseline algorithm	Mean Rank	%acc@500/1k/ 5k/10k	%match
ADD	29912	1.7/5.1/8.5/16.2	48
MUL	62601	0.0/1.7/4.2/5.9	49

Future Work

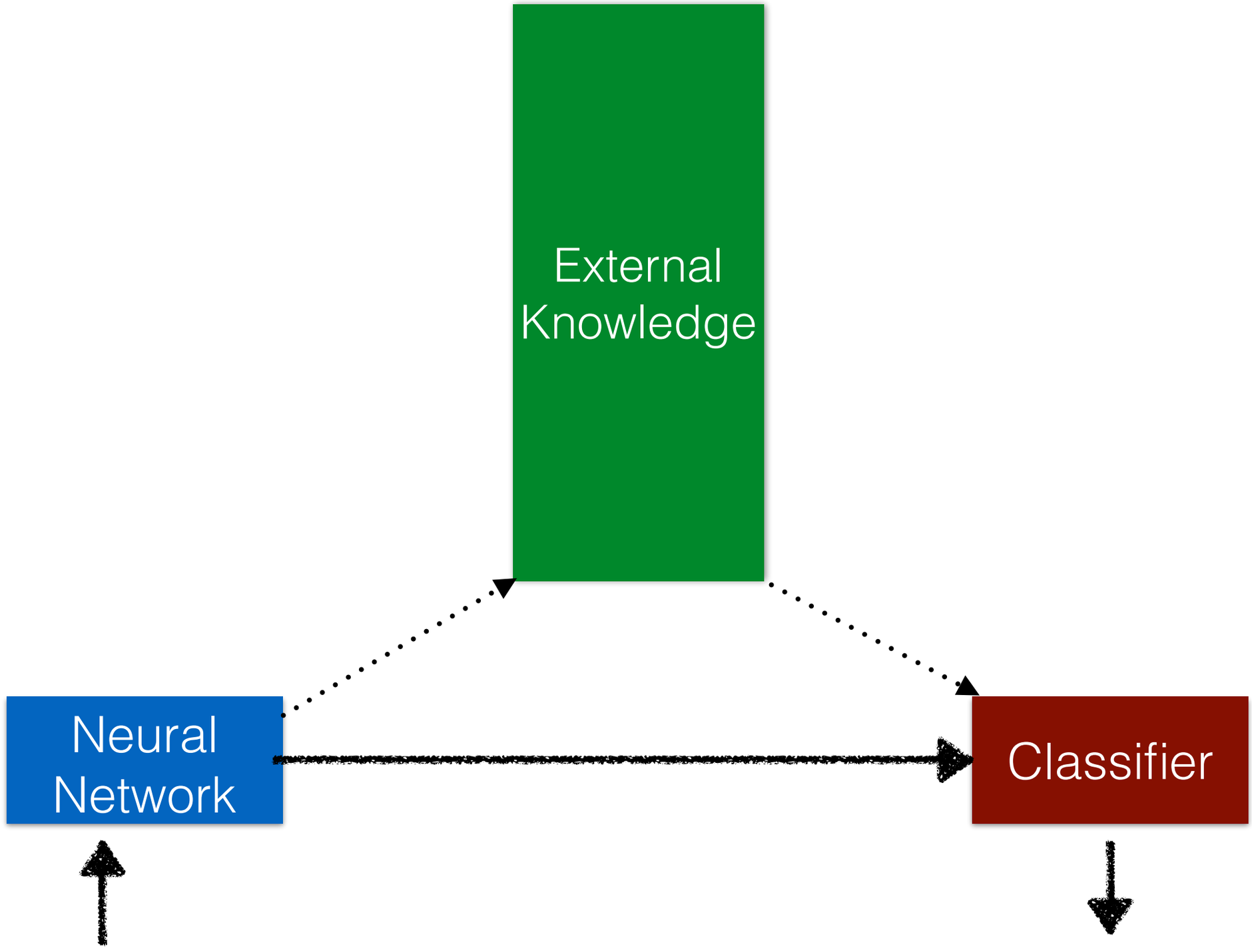
- Use pre-trained word embeddings from spaCy to improve the baseline performance.
- Implement a RNN model to learn the word embeddings and compare the performance with respect to the baseline methods.

I. Felix Hill, Kyunghyun Cho, Anna Korhonen and Yoshua Bengio. Learning to understand phrases by embedding the dictionary. *Association for Computational Linguistics*, vol 4, 2016.
II. http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Who wrote the song "Kiss from a Rose"?



Can we replace all of these modules with a single neural network?



- **factoid** QA: the answer is a single entity / numeric
  - “who wrote the book “Dracula”?
- **non-factoid** QA: answer is free text
  - “why is Dracula so evil?”
- **QA subtypes (could be factoid or non-factoid):**
  - **semantic parsing:** question is mapped to a logical form which is then executed over some database
    - “how many people did Dracula bite?”
- **reading comprehension**: answer is a span of text within a document (could be factoid or non-factoid)
- **community-based QA:** question is answered by multiple web users (e.g., Yahoo! Answers)
- **visual QA:** questions about images

Machine reading
(“reading comprehension”)

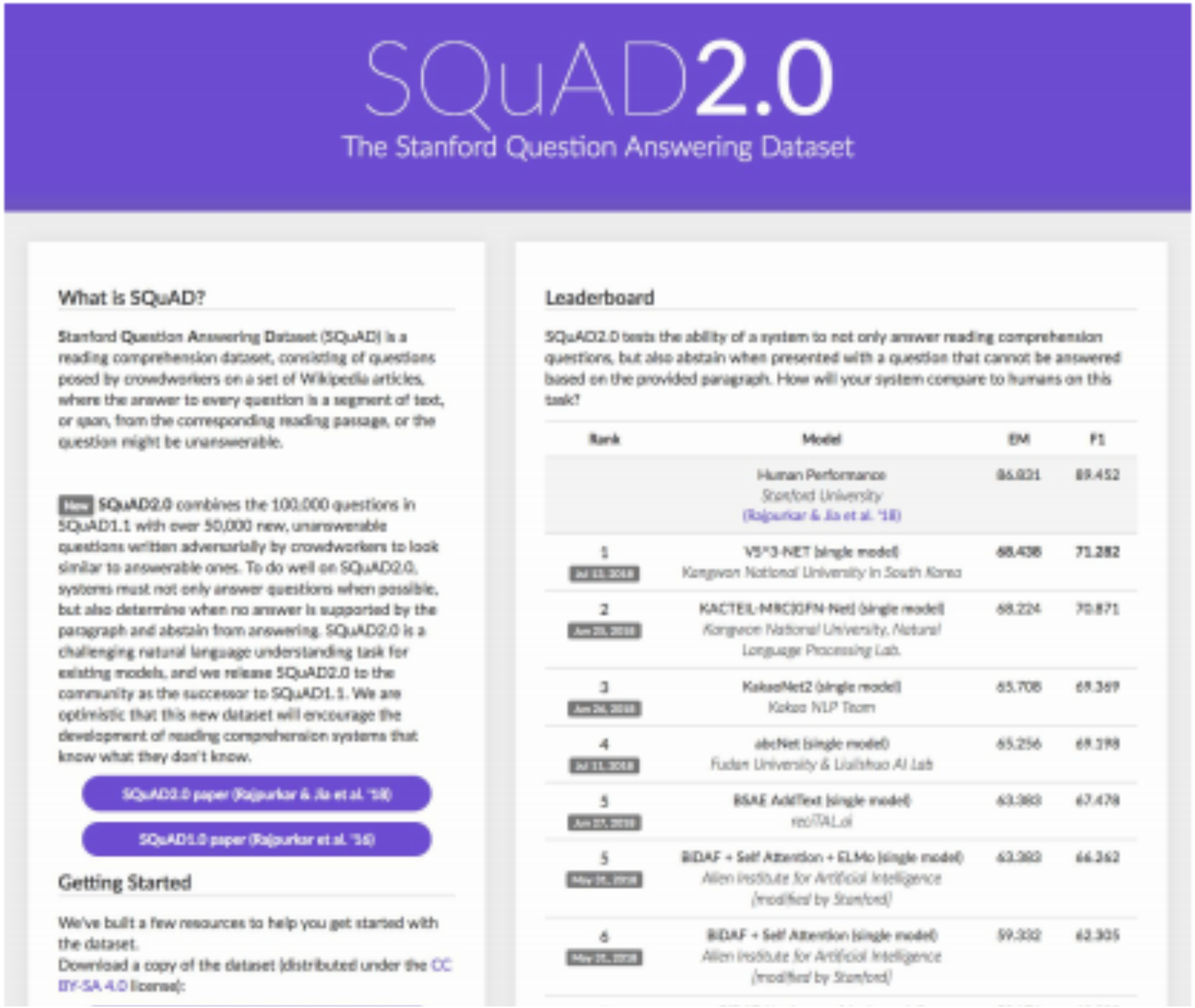
Narrative QA

Narrative QA: examples

Question: How is Oscar related to Dana?
Answer: He is her son

Summary snippet: ...Peter's former girlfriend Dana Barrett has had a son, Oscar...
Story snippet:
DANA (setting the wheel brakes on the buggy) Thank you, Frank. I'll get the hang of this eventually.
She continues digging in her purse while Frank leans over the buggy and makes funny faces at the baby, OSCAR, a very cute nine-month old boy.
FRANK (to the baby) Hiya, Oscar. What do you say, slugger?
FRANK (to Dana) That's a good-looking kid you got there, Ms. Barrett.

SQuAD



SQuAD2.0

The Stanford Question Answering Dataset

What is SQuAD?

Stanford Question Answering Dataset (SQuAD) is a reading comprehension dataset, consisting of questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage, or the question might be unanswerable.

New SQuAD2.0 combines the 100,000 questions in SQuAD1.1 with over 50,000 new, unanswerable questions written adversarially by crowdworkers to look similar to answerable ones. To do well on SQuAD2.0, systems must not only answer questions when possible, but also determine when no answer is supported by the paragraph and abstain from answering. SQuAD2.0 is a challenging natural language understanding task for existing models, and we release SQuAD2.0 to the community as the successor to SQuAD1.1. We are optimistic that this new dataset will encourage the development of reading comprehension systems that know what they don't know.

SQuAD2.0 paper (Rajpurkar & Jia et al. '18)

SQuAD1.0 paper (Rajpurkar et al. '16)

Getting Started

We've built a few resources to help you get started with the dataset.
Download a copy of the dataset (distributed under the CC BY-SA 4.0 license):

Leaderboard

SQuAD2.0 tests the ability of a system to not only answer reading comprehension questions, but also abstain when presented with a question that cannot be answered based on the provided paragraph. How will your system compare to humans on this task?

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar & Jia et al. '18)	86.831	89.452
1 Jul 11, 2018	V5*3-NET (single model) Kangwon National University in South Korea	68.438	71.282
2 Jun 28, 2018	KACTEIL-MRC(DFN-Net) (single model) *Kangwon National University, Natural Language Processing Lab.*	68.224	70.871
3 Jun 26, 2018	KakaoNet2 (single model) Kakao NLP Team	65.708	69.369
4 Jul 11, 2018	abcNet (single model) Fudan University & Liulishuo AI Lab	65.256	69.198
5 Jun 23, 2018	B&AE AdaText (single model) rezTAL.ai	63.380	67.478
5 May 31, 2018	BiDAF + Self Attention + ELMo (single model) *Allen Institute for Artificial Intelligence (modified by Stanford)*	63.380	66.262
6 May 31, 2018	BiDAF + Self Attention (single model) *Allen Institute for Artificial Intelligence (modified by Stanford)*	59.332	62.305

SQuAD

In the 1960s, a series of discoveries, the most important of which was seafloor spreading, showed that the Earth's lithosphere, which includes the crust and rigid uppermost portion of the upper mantle, is separated into a number of tectonic plates that move across the plastically deforming, solid, upper mantle, which is called the asthenosphere. There is an intimate coupling between the movement of the plates on the surface and the convection of...

Question:
Which parts of the Earth are included in the lithosphere?

How would you go about building a model for SQuAD?

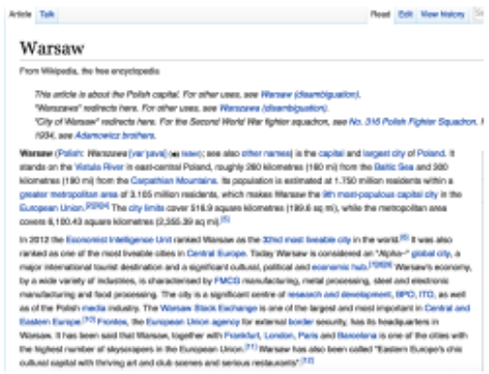
Let’s look at the DRQA model (Chen et al., ACL 2017)

Overview of the Document Reader Question Answering

Q: How many of Warsaw's inhabitants spoke Polish in 1933?

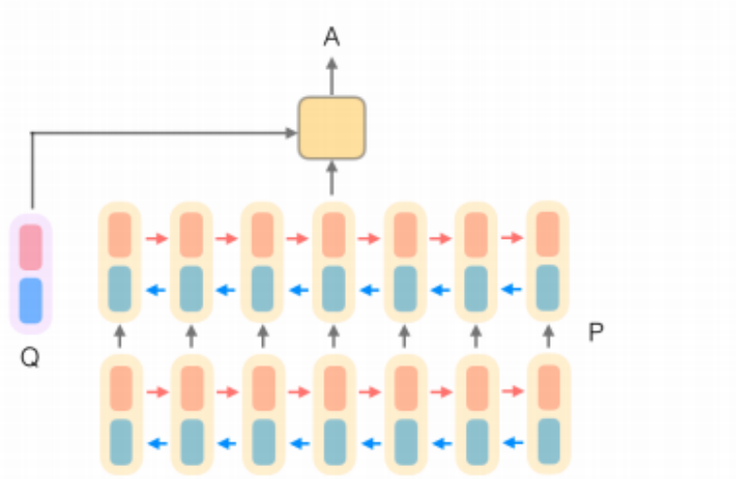
WIKIPEDIA
The Free Encyclopedia

Document Retriever



Document Reader

833,500



Good source code available!

Big idea

Super Bowl 50 was an American football game to determine the champion of the National Football League (NFL) for the 2015 season. The American Football Conference (AFC) champion Denver Broncos defeated the National Football Conference (NFC) champion Carolina Panthers 24–10 to earn their third Super Bowl title. The game was played on February 7, 2016, at Levi's Stadium in the San Francisco Bay Area at Santa Clara, California. As this was the 50th Super Bowl, the league emphasized the "golden anniversary" with various gold-themed initiatives, as well as temporarily suspending the tradition of naming each Super Bowl game with Roman numerals (under which the game would have been known as "Super Bowl L"), so that the logo could prominently feature the Arabic numerals 50.

Q: Which NFL team represented the AFC at Super Bowl 50?

A: Denver Broncos

Start and End Probabilities

$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\} \tag{1}$$

$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\} \tag{2}$$

1. A vector representing our question
2. Vector representing each word in the query text
3. Parameter: here's the start/end of the answer

Start and End Probabilities

$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\} \tag{1}$$

$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\} \tag{2}$$

1. A vector representing our question
2. Vector representing each word in the query text
3. Parameter: here's the start/end of the answer

Start and End Probabilities

$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\} \tag{1}$$

$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\} \tag{2}$$

1. A vector representing our question
2. Vector representing each word in the query text
3. Parameter: here's the start/end of the answer

Start and End Probabilities

$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\} \quad (1)$$

$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\} \quad (2)$$

1. A vector representing our question
2. Vector representing each word in the query text
3. Parameter: here's the start/end of the answer

Start and End Probabilities

$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\} \tag{1}$$

$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\} \tag{2}$$

1. A vector representing our question
2. Vector representing each word in the query text
3. Parameter: here's the start/end of the answer

Start and End Probabilities

$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\} \quad (1)$$

$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\} \quad (2)$$

1. A vector representing our question
2. Vector representing each word in the query text
3. Parameter: here's the start/end of the answer

Other ways of modeling possible answers?

Start and End Probabilities

$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\} \quad (1)$$

$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\} \quad (2)$$

1. A vector representing our question
2. Vector representing each word in the query text
3. Parameter: here's the start/end of the answer

How does this work at test-time?

Question Encoding

$$\vec{q} = \sum_j b_j \vec{q}_j \tag{3}$$

$$b_j = \frac{\exp\left\{\vec{w} \cdot q_j\right\}}{\sum_{j'} \exp\left\{w \cdot q_{j'}\right\}} \tag{4}$$

Question Encoding

$$\vec{q} = \sum_j b_j \vec{q}_j \tag{3}$$

$$b_j = \frac{\exp\left\{\vec{w} \cdot q_j\right\}}{\sum_{j'} \exp\left\{w \cdot q_{j'}\right\}} \tag{4}$$

Question vector is a weighted sum

Question Encoding

$$\vec{q} = \sum_j b_j \vec{q}_j \tag{3}$$

$$b_j = \frac{\exp\left\{\vec{w} \cdot q_j\right\}}{\sum_{j'} \exp\left\{w \cdot q_{j'}\right\}} \tag{4}$$

The weight is a scalar

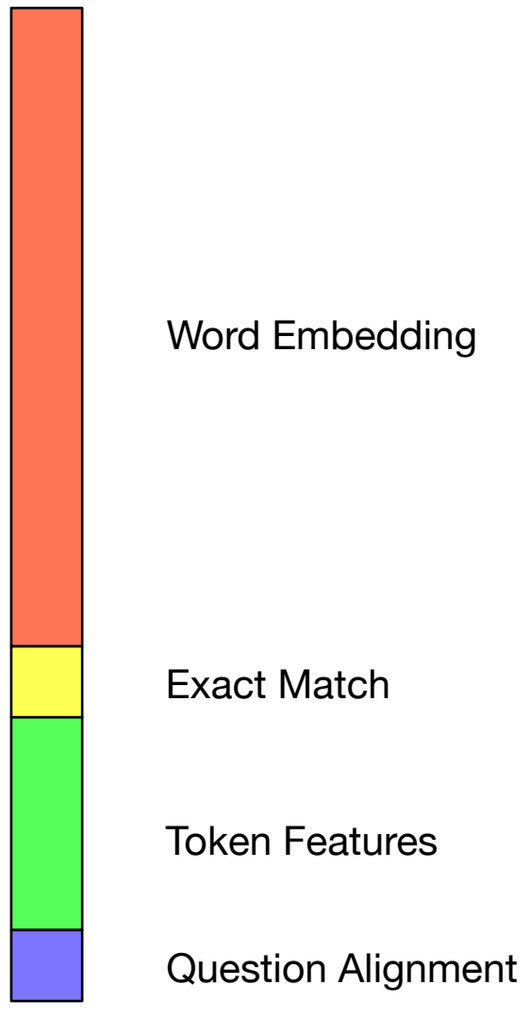
Question Encoding

$$\vec{q} = \sum_j b_j \vec{q}_j \tag{3}$$

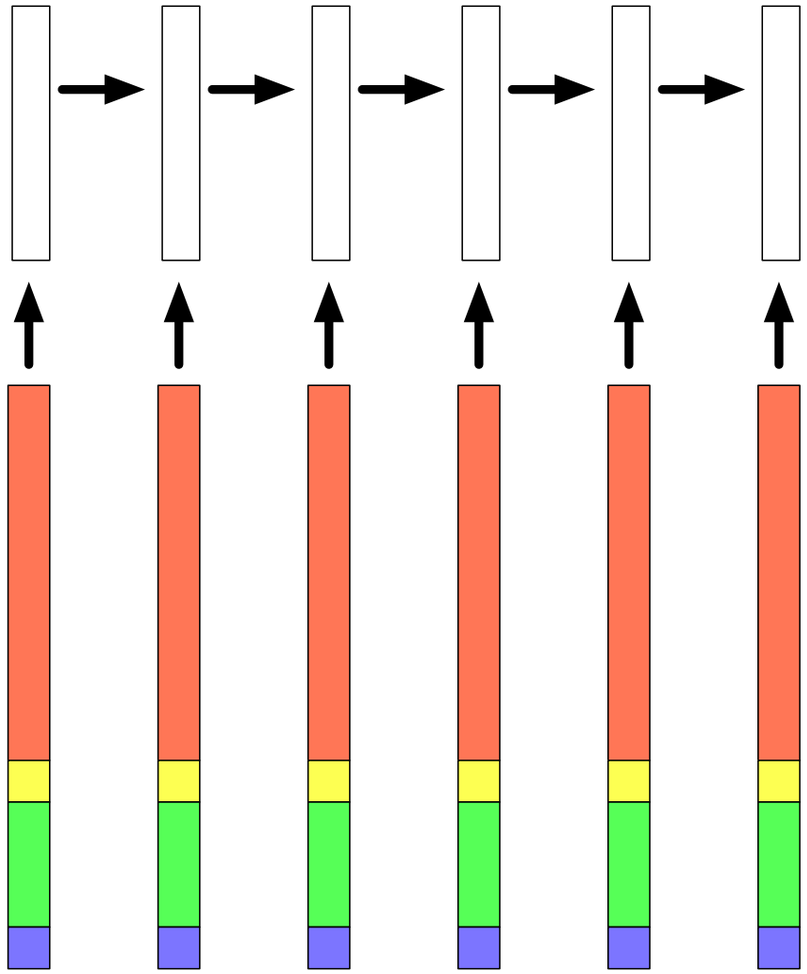
$$b_j = \frac{\exp\left\{\vec{w} \cdot q_j\right\}}{\sum_{j'} \exp\left\{w \cdot q_{j'}\right\}} \tag{4}$$

A focus parameter learns how to focus on particular words in the question

Paragraph Encoding

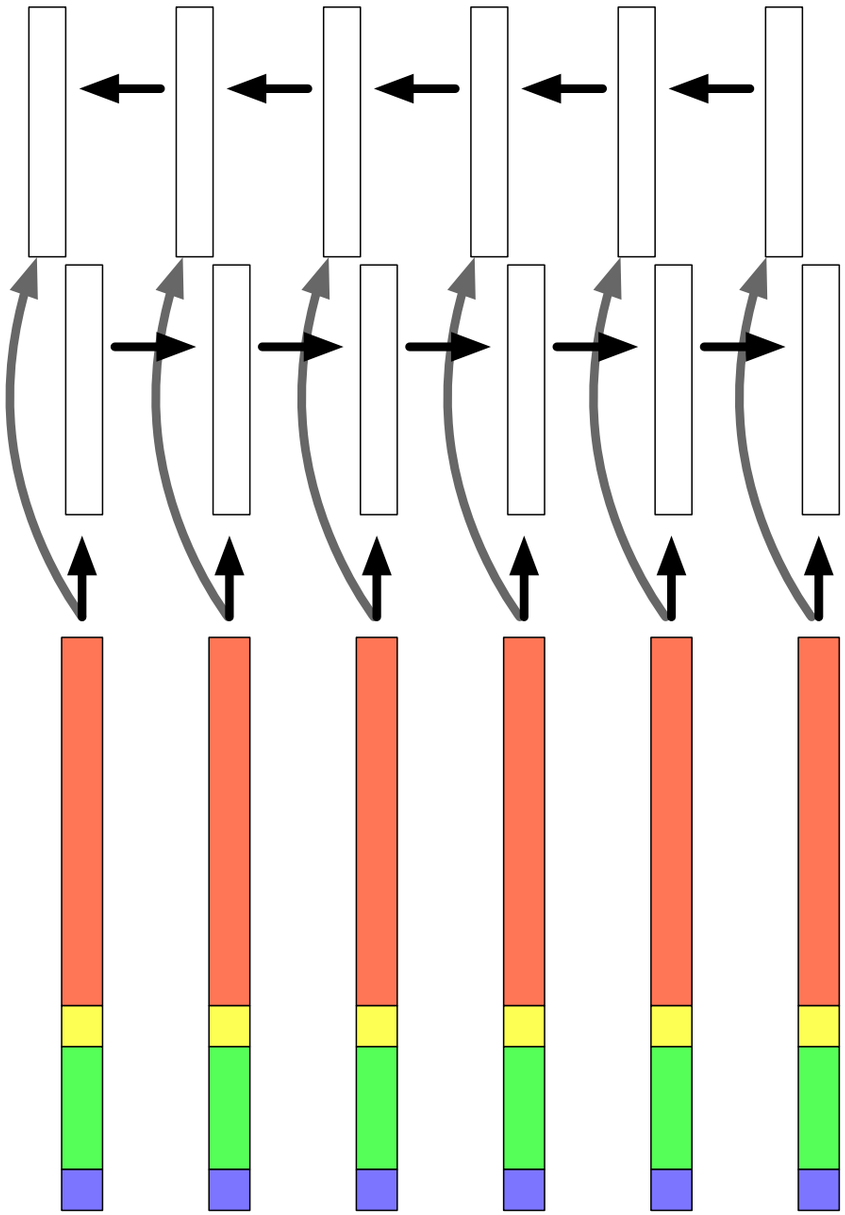


Paragraph Encoding



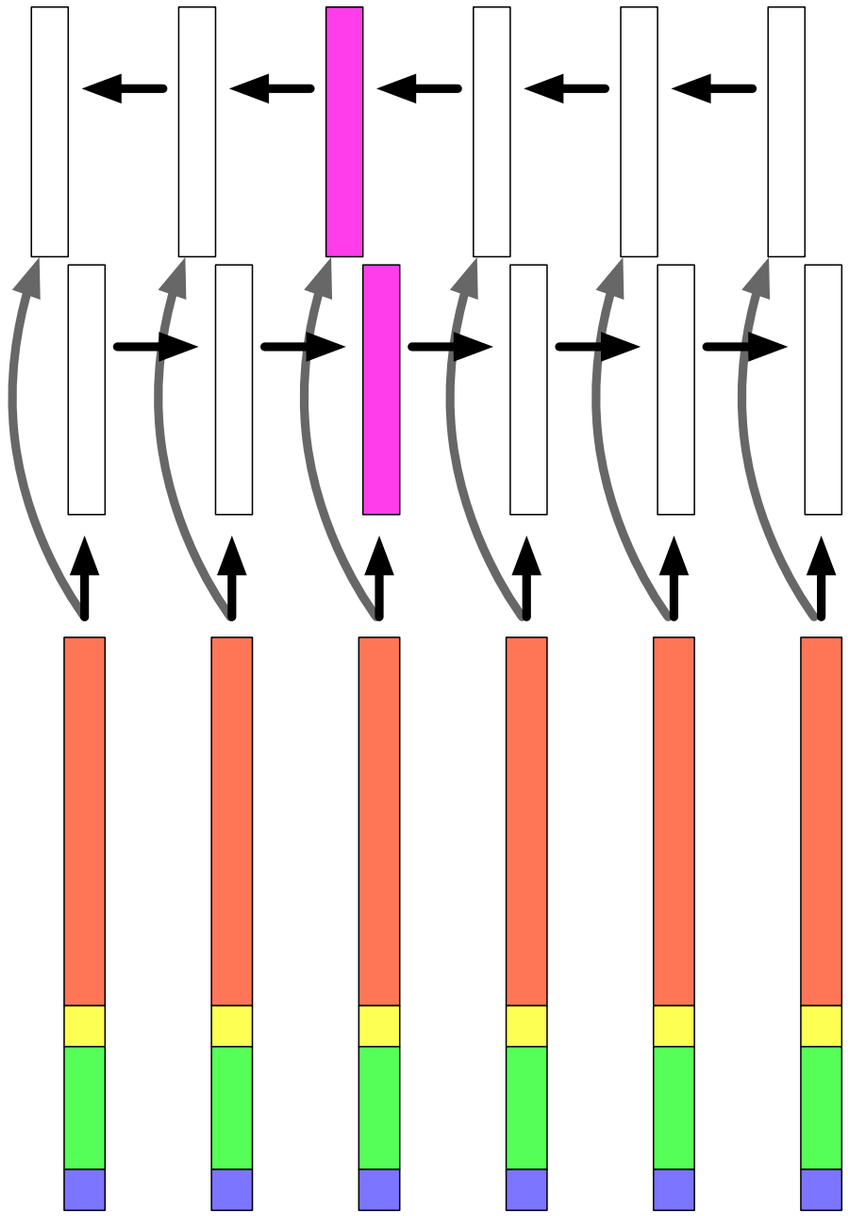
LSTM: encode
contextual effects

Paragraph Encoding



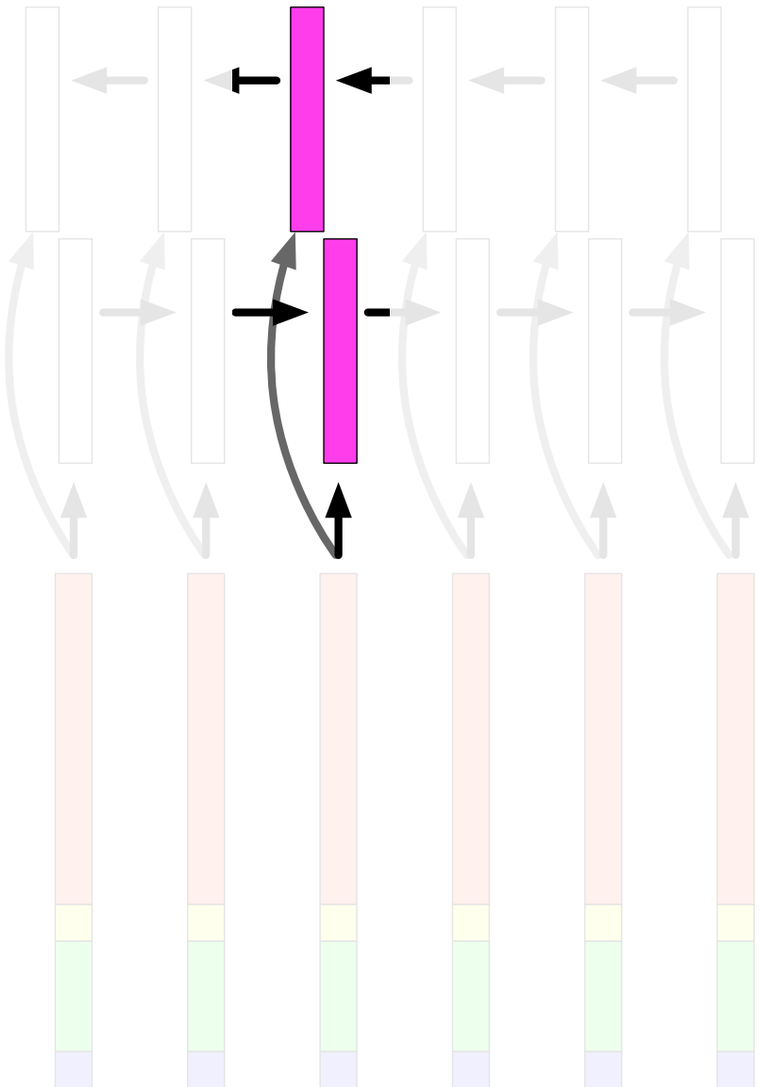
Add a backwards direction as well (bi-directional LSTM)

Paragraph Encoding



Use the concatenation of these two hidden layers as the representation of the word

Paragraph Encoding



$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\}$$

$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\}$$

Implementation

- Trained on passages
- Backprop through all layers
- Look at code

```
# RNN document encoder
self.doc_rnn = layers.StackedBRNN(
    input_size=doc_input_size,
    hidden_size=args.hidden_size,
    num_layers=args.doc_layers,
    dropout_rate=args.dropout_rnn,
    dropout_output=args.dropout_rnn_output,
    concat_layers=args.concat_rnn_layers,
    rnn_type=self.RNN_TYPES[args.rnn_type],
    padding=args.rnn_padding,
)
```

```
# RNN question encoder
self.question_rnn = layers.StackedBRNN(
    input_size=args.embedding_dim,
    hidden_size=args.hidden_size,
    num_layers=args.question_layers,
    dropout_rate=args.dropout_rnn,
    dropout_output=args.dropout_rnn_output,
    concat_layers=args.concat_rnn_layers,
    rnn_type=self.RNN_TYPES[args.rnn_type],
    padding=args.rnn_padding,
)
```

https://github.com/facebookresearch/DrQA/

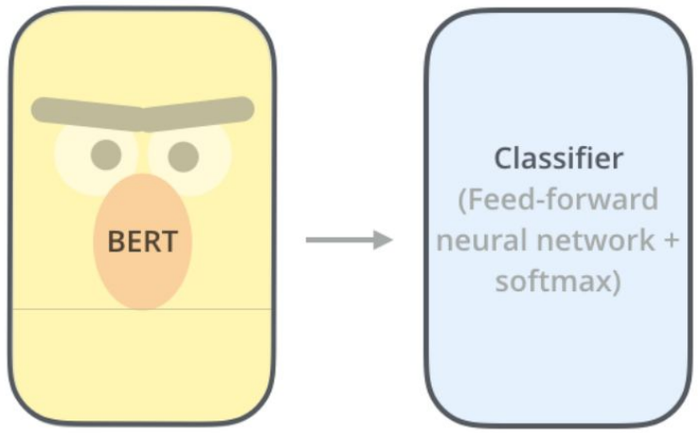
Thieves of Sesame Street: Model Extraction on BERT-based APIs

@kalpeshk
with @gtomar, @aparikh

UMass Amherst

Google AI

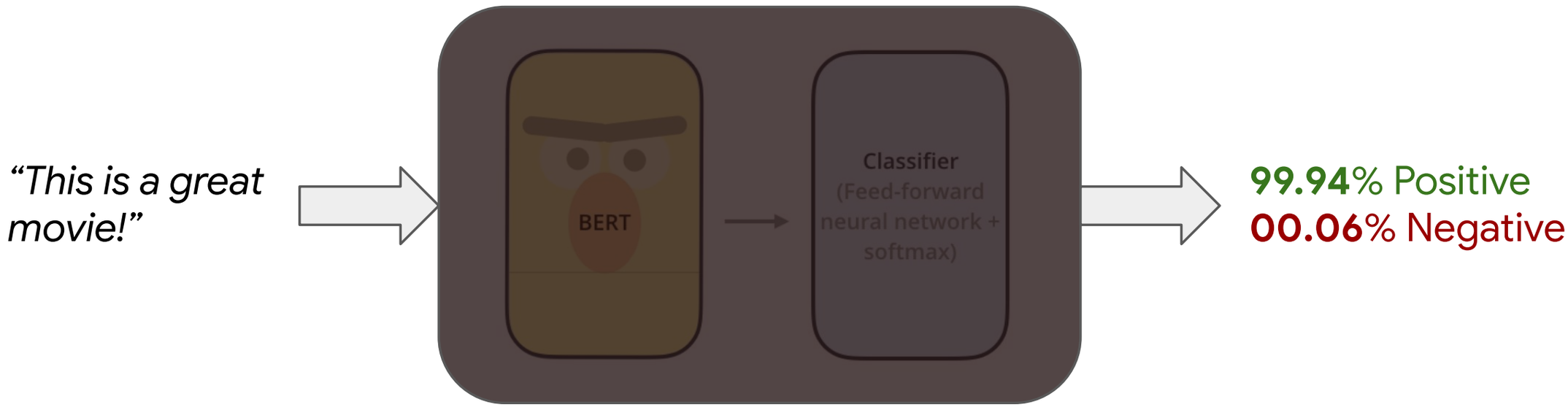
What is model extraction?



A company trains a BERT-based textual classifier

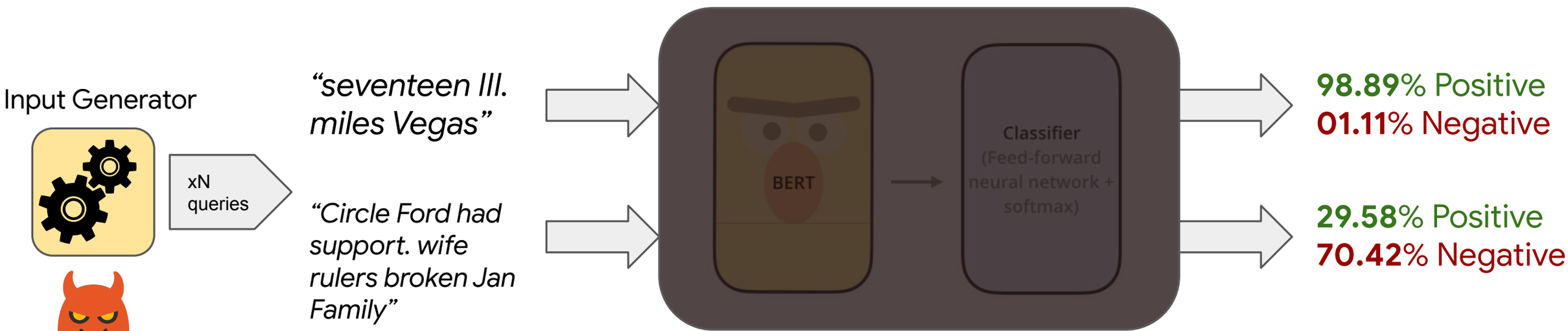
Photo credits - http://jalamm

What is model extraction?



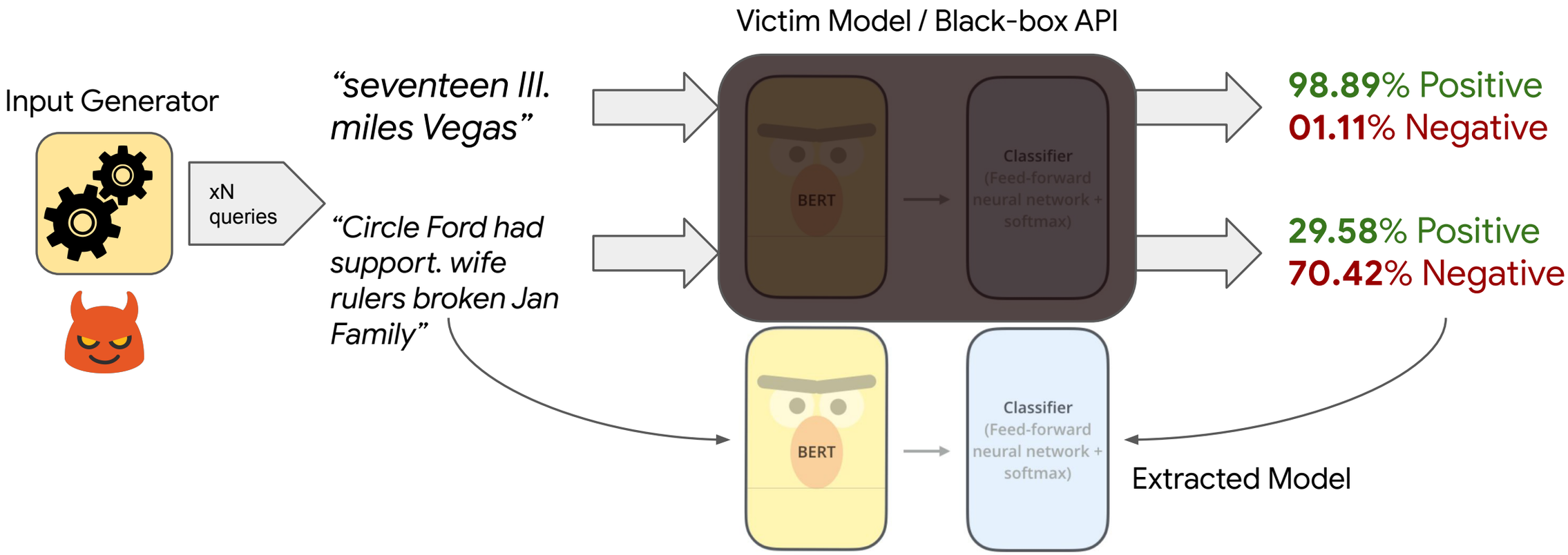
Releases it as a Cloud API, with black-box query access

What is model extraction?



Malicious user generates inputs and spams API

What is model extraction?



API outputs used as training data

Step 1: Attacker randomly samples words to form queries and sends them to victim BERT model

passage 1: before selling ?' New about to in Week the American each Colonel characters, from and as in including and a shooter Efforts happened, as on as measured. and and the (which proper and that as Ric for living interest Air …

question: During and living and in selling Air?

passage 2: Arab in (Dodd) singer, as to orthologues November giving small screw Peng be at and sea national Fire) there to support south Classic, Quadrille promote filmed …

question: Which national giving Classic, Quadrille national as?

Victim model (blackbox API)

BERT

Feed-forward classifier for fine-tuning

Step 2: Attacker fine-tunes their own BERT on these queries using the victim outputs as labels

Victim output 1: Ric

Victim output 2: south Classic

BERT

Feed-forward classifier for fine-tuning

Extracted model

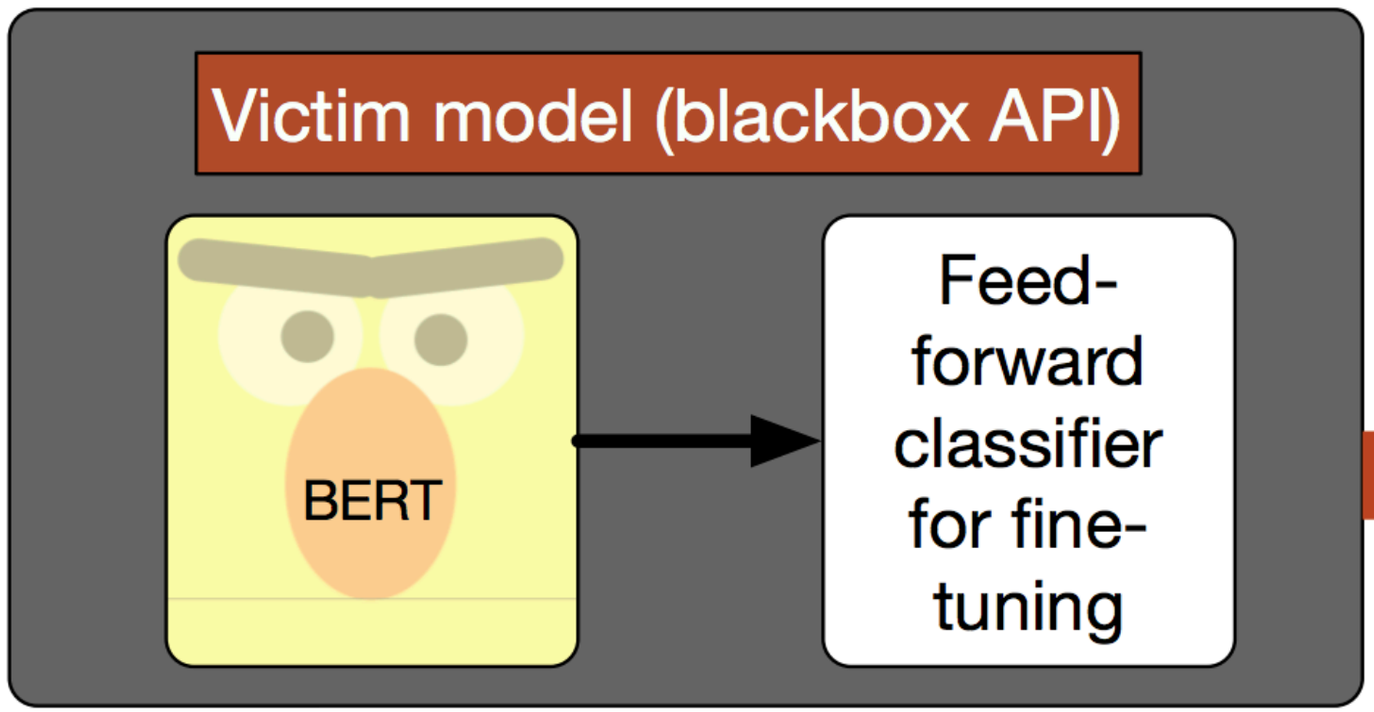
Step 1: Attacker randomly samp
words to form queries and send
them to victim BERT model

passage 1: before selling ?' New about to in Week the American each Colonel characters, from and as in including and a shooter Efforts happened, as on as measured. and and the (which proper and that as Ric for living interest Air …

question: During and living and in selling Air?

passage 2: Arab in (Dodd) singer, as to orthologues November giving small screw Peng be at and sea national Fire) there to support south Classic, Quadrille promote filmed …

question: Which national giving Classic, Quadrille national as?



Step 2: Attacker fine-tunes their own BERT on these queries using the victim outputs as labels

Victim output 1: Ric

Victim output 2: south Classic

Task	RANDOM example
SST2	cent 1977, preparation (120 remote Program finance add broader protection (**76.54**% negative)
MNLI	**P**: Mike zone fights Woods Second State known, defined come **H**: Mike zone released, Woods Second HMS males defined come (**99.89**% contradiction)
SQuAD	**P**: a of Wood, curate him and the ” Stop Alumni terrestrial the of of roads Kashyap. Space study with the Liverpool, Wii Jordan night Sarah lbf a Los the Australian three English who have that that health officers many new workforce... **Q**: *How workforce. Stop who new of Jordan et Wood, displayed the?* **A**: **Alumni terrestrial the of of roads Kashyap**

WIKI example

So many were produced that thousands were Brown's by coin 1973 (**98.59**% positive)

P: voyage have used a variety of methods to Industrial their Trade
H: descent have used a officially of methods exhibition Industrial their Trade (**99.90**% entailment)

P: Since its release, Dookie has been featured heavily in various "must have" lists compiled by the music media. Some of the more prominent of these lists to feature Dookie are shown below; this information is adapted from Acclaimed Music.
Q: *What are lists feature prominent " adapted Acclaimed are various information media.?*
A: **"must have"**

Economically practical?

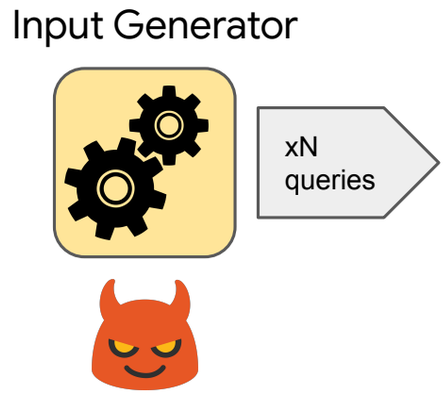
- Google Cloud NL API, <= $1.00 for every 1000 API calls.
- Lots of free schemes, distributed collection possible

Dataset	**Size**	**Upperbound Price**
SST2 (sentiment)	67349 sentences	$62.35
Switchboard (speech)	300 hours	$430.56
MNLI (pairwise inference)	392702 pairs	$387.70
Translation	1 million sentences (100 characters each)	$2000.00

https://cloud.google.com/products/ca

Simple Attacks

1. RANDOM
    a. Randomly sample word sequences
    b. Apply task-specific heuristic
2. WIKI
    a. Collect wikitext103 sentences
    b. Apply task-specific heuristic



In most cases inputs are nonsensical to humans.

Simple Attacks - Results

	Setting		*API Dev%*	*Extracted Models Dev%*		
Task	**Queries**	**Cost**	**ORIGINAL**	**RANDOM**	**WIKI**	**WIKI-ARGMAX**
SST-2	67349	$62.35	93.12%	90.06%	91.40%	91.28%
MNLI	392702	$387.70	85.80%	76.26%	77.80%	77.12%
SQuAD	87599	$82.60	90.58 F1	79.61 F1	86.20 F1	-
BoolQ	9427	$4.43	76.13%	X	66.78%	66.04%
BoolQ (50x)	471350	$466.35	76.13%	-	72.71%	-

Extraction is quite effective, even with
out-of-distribution input points!

Simple Attacks - Results

	Setting		*API Dev%*	*Extracted Models Dev%*		
Task	**Queries**	**Cost**	**ORIGINAL**	**RANDOM**	**WIKI**	**WIKI-ARGMAX**
SST-2	67349	$62.35	93.12%	90.06%	91.40%	91.28%
MNLI	392702	$387.70	85.80%	76.26%	77.80%	77.12%
SQuAD	87599	$82.60	90.58 F1	79.61 F1	86.20 F1	-
BoolQ	9427	$4.43	76.13%	X	66.78%	66.04%
BoolQ (50x)	471350	$466.35	76.13%	-	72.71%	-

Extraction is quite effective, even with out-of-distribution input points!

Simple Attacks - Results

Setting			*API Dev%*	*Extracted Models Dev%*		
Task	**Queries**	**Cost**	**ORIGINAL**	**RANDOM**	**WIKI**	**WIKI-ARGMAX**
SST-2	67349	$62.35	93.12%	90.06%	91.40%	91.28%
MNLI	392702	$387.70	85.80%	76.26%	77.80%	77.12%
SQuAD	87599	$82.60	90.58 F1	79.61 F1	86.20 F1	-
BoolQ	9427	$4.43	76.13%	X	66.78%	66.04%
BoolQ (50x)	471350	$466.35	76.13%	-	72.71%	-

Extraction is quite effective, even with out-of-distribution input points!

Extraction improves with more queries

(89.4 F1 with $826 on SQuAD
vs 90.6 F1 with original API)

Quiz Bowl

what is quiz bowl?

- a trivia game that contains questions about famous entities (e.g., novels, battles, countries)

- developed a deep learning system, QANTA, to play quiz bowl

- one of the first applications of deep learning to question answering

Iyyer et al., EMNLP 2014 & ACL 2015

This author described a "plank in reason" breaking and hitting a "world at every plunge" in a poem which opens "I felt a funeral in my brain."

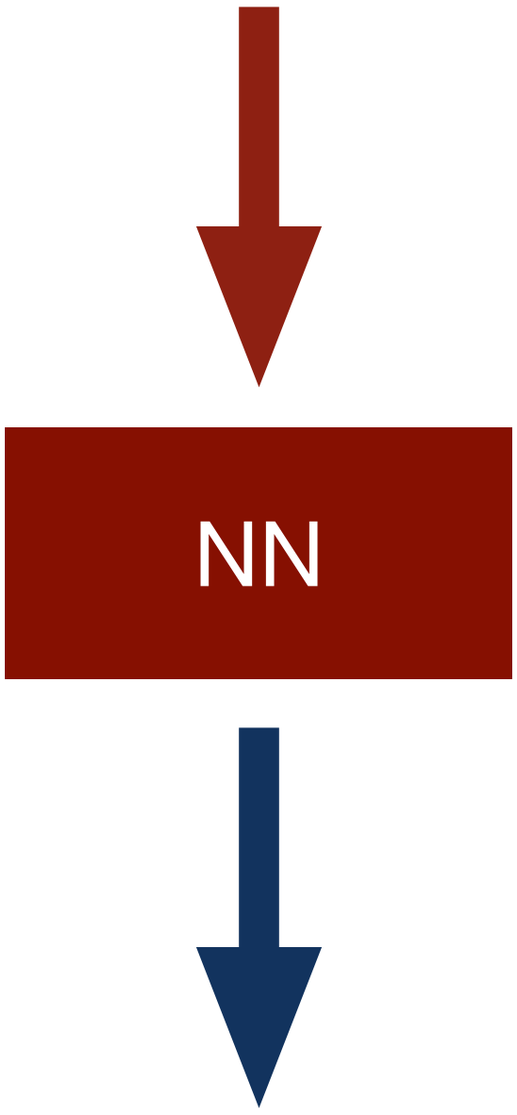
She wrote that "the stillness round my form was like the stillness in the air" in "I heard a fly buzz when I died."

She wrote about a scarcely visible roof and a cornice that was "but a mound" in a poem about a carriage ride with Immortality and Death.

For 10 points, name this reclusive "Belle of Amherst" who wrote "Because I could not stop for Death."

A: Emily Dickinson

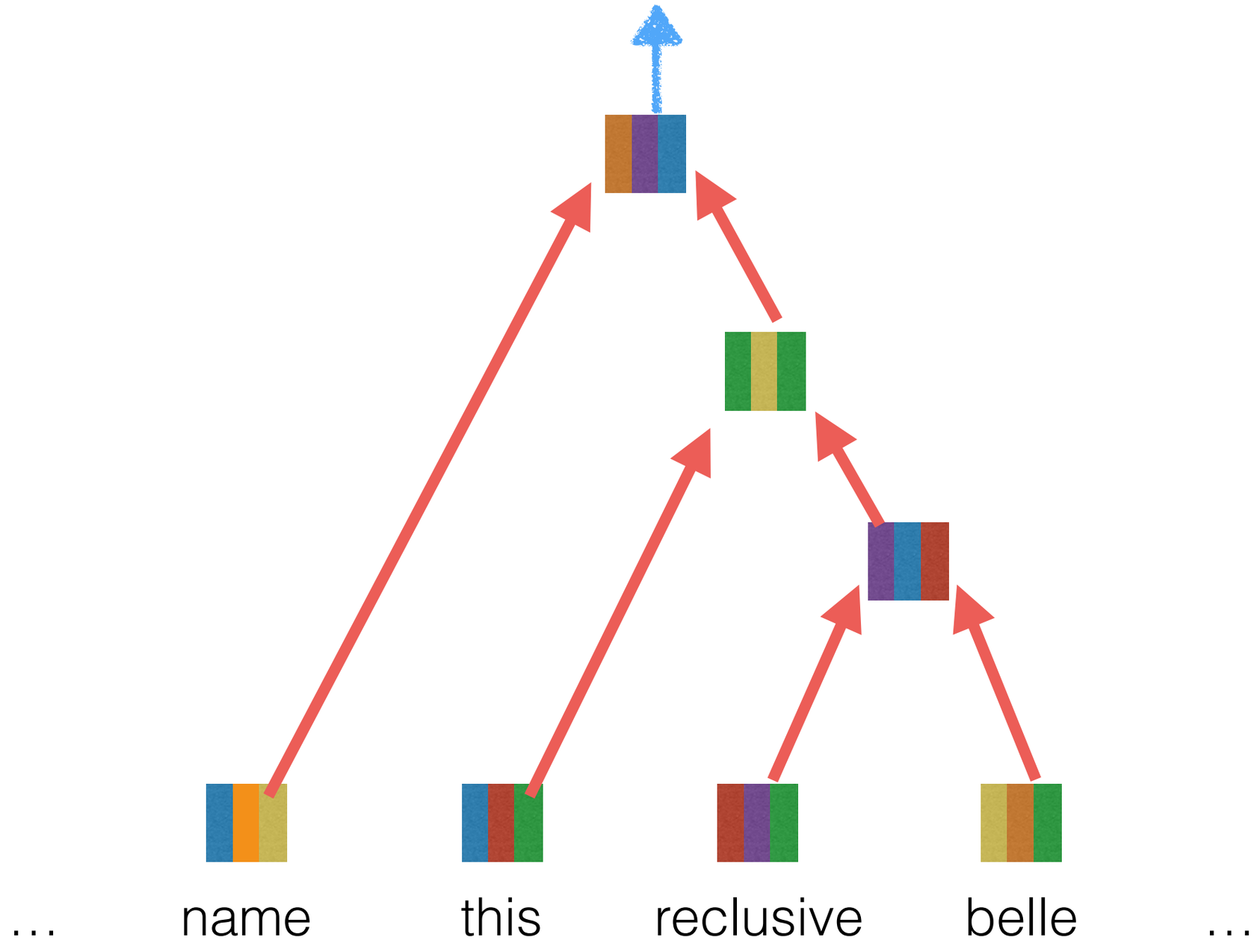
… name this reclusive "Belle of Amherst”…



Emily Dickinson

dependency-tree NNs

softmax: predict **Emily Dickinson** out of a set of ~5000 answers



Iyyer et al., EMNLP 2014

simple discourse-level representations by averaging

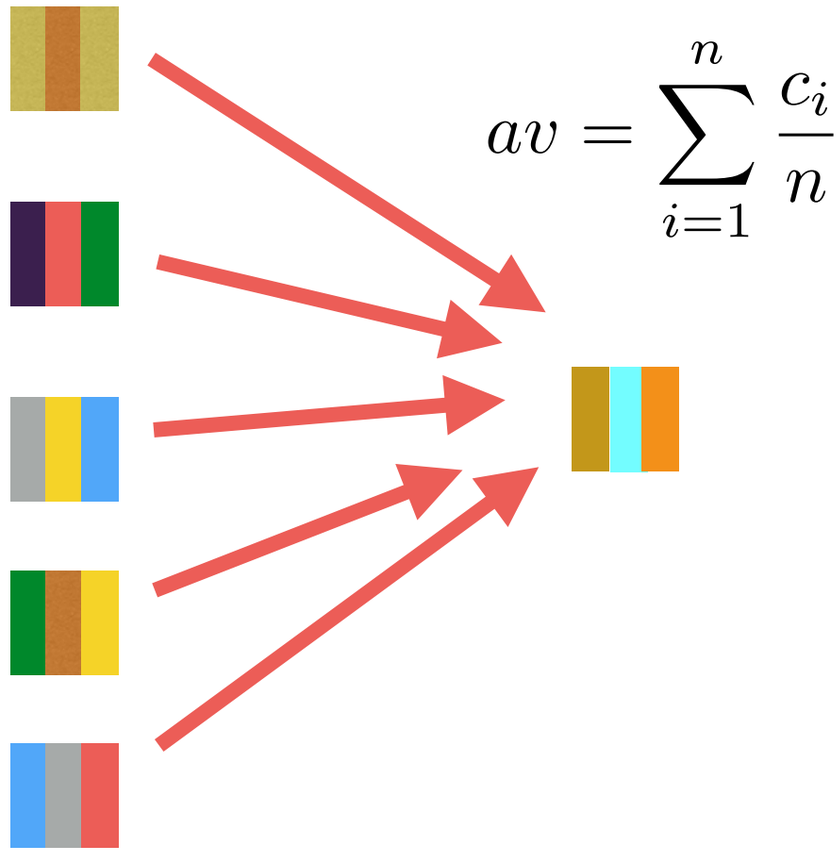
In one novel, one of these figures antagonizes an impoverished family before leaping into an active volcano.

Another of these figures titles a novella in which General Spielsdorf describes the circumstances of his niece Bertha Reinfeldt's death to the narrator, Laura.

In addition to Varney and Carmilla, another of these figures sails on the Russian ship Demeter in order to reach London.

That figure bites Lucy Westenra before being killed by a coalition including Jonathan Harker and Van Helsing.

For 10 points, identify these bloodsucking beings most famously exemplified by Bram Stoker’s Dracula.

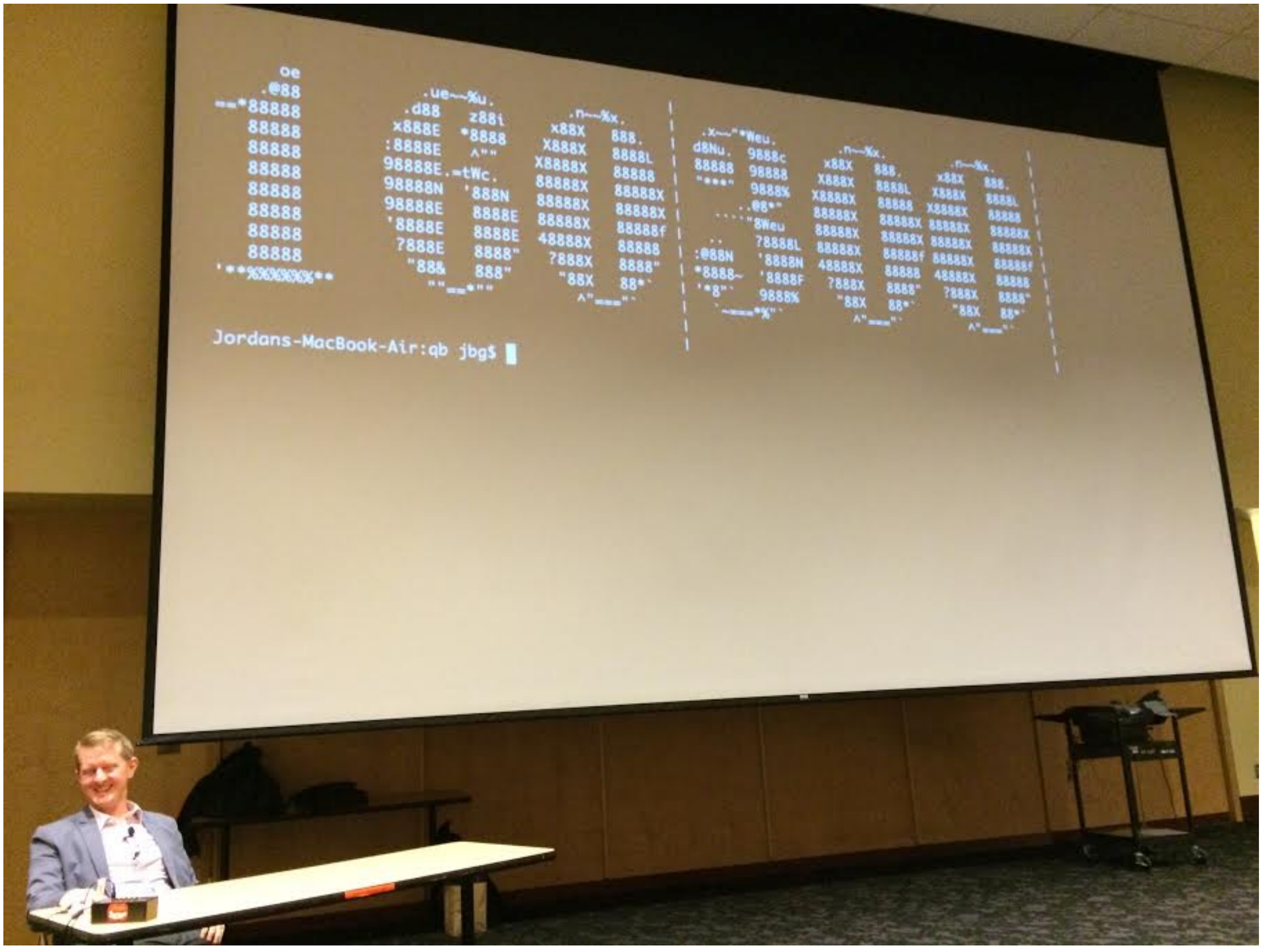


comparison of architectures

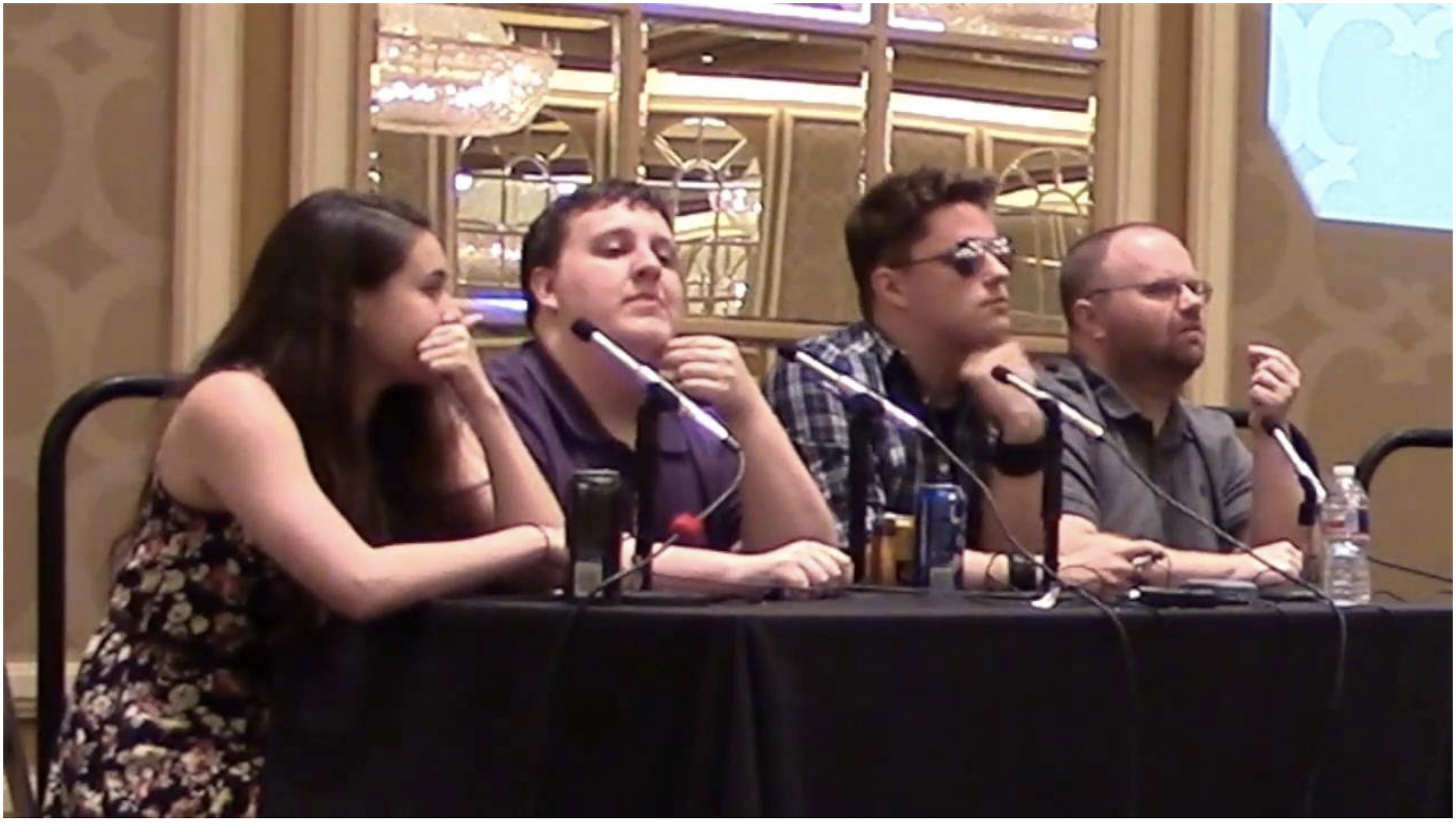
Model	**Quizbowl Accuracy**	**Time / Epoch (s)**
NBOW	66.3	11
DAN	70.8	18
Tree-NN	72.1	314

similar results have since been shown for other tasks such as entailment and sentence similarity (Wieting et al., ICLR 2016, Hill et al., NAACL 2016)

2015: defeated Ken Jennings 300-160

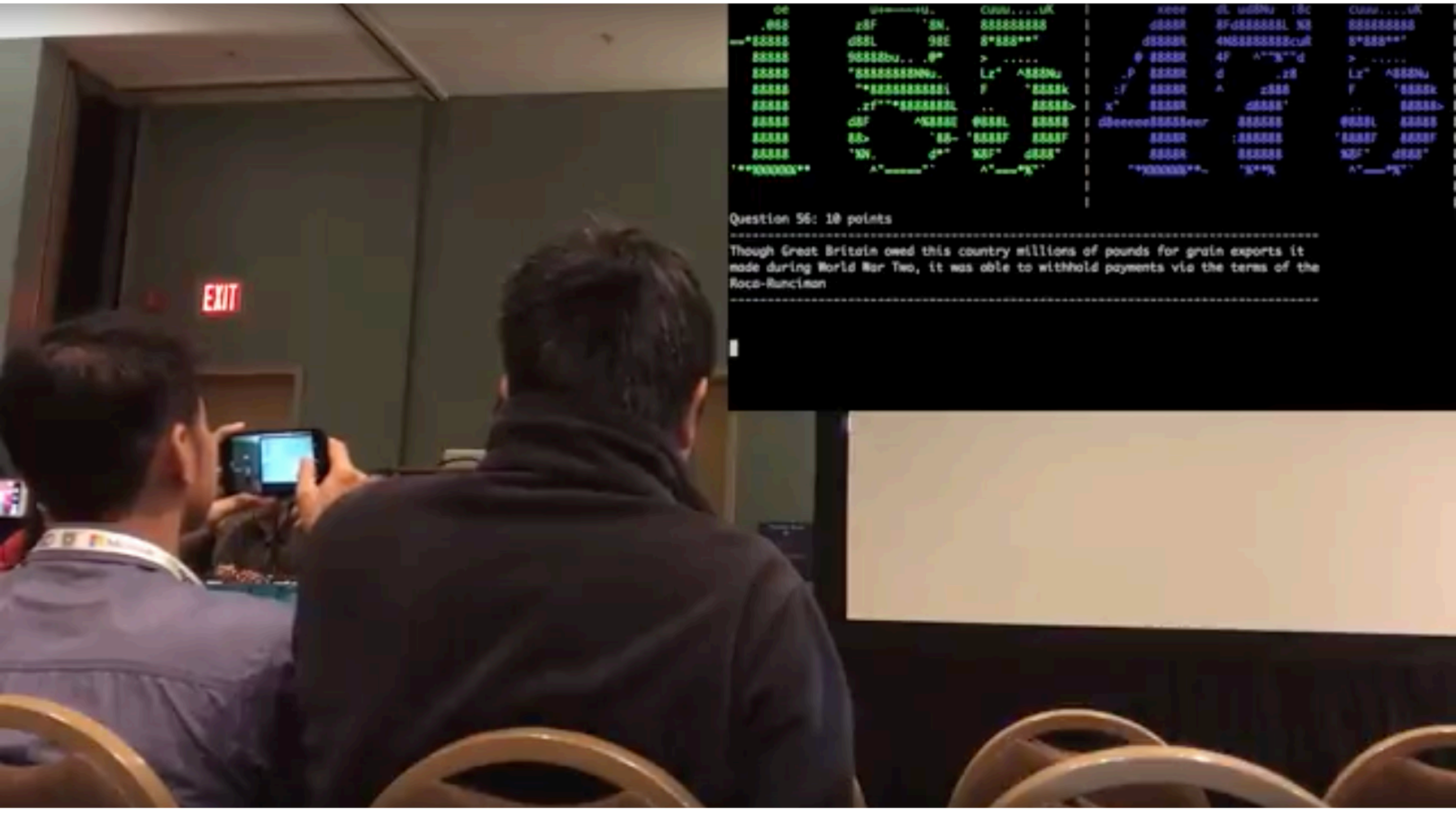


2016: lost to top quiz bowlers 345-145



2017: beat top quiz bowlers 260-215

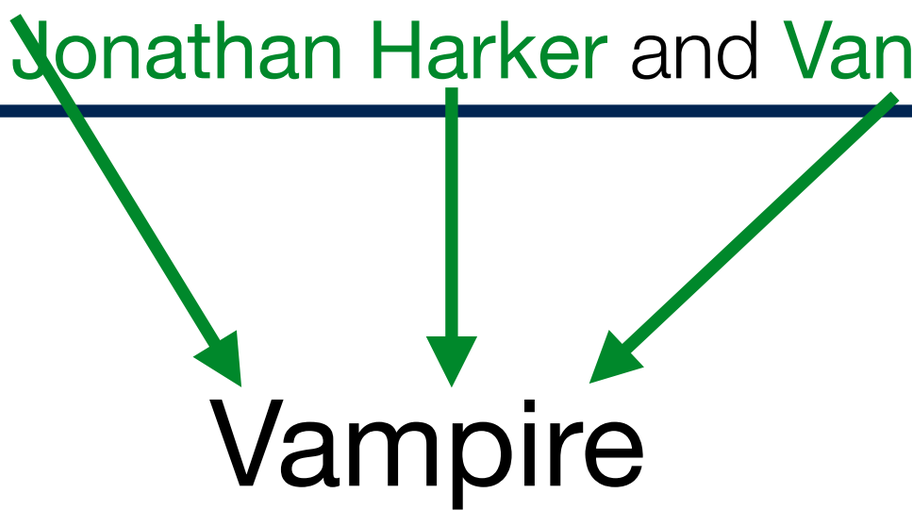
late 2017: crushed top team 475-185



deep learning ~ memorization

during training, **QANTA** becomes very good at associating named entities in questions with answers…

That figure bites Lucy Westenra before being killed by a coalition including Jonathan Harker and Van Helsing.



deep learning ~ memorization

during training, QANTA becomes very good at associating named entities in questions with answers…

In one novel, one of these figures antagonizes an impoverished family before leaping into an active volcano.

???