NPs in RNNs
Dori Sharabani and Lihi Nahir
*Himself ate the cake.

He ate the cake.

‘he’ – pronoun
‘himslef’ – reflexive pronoun (anaphor)
John asked Mary to talk to himself.

John asked Mary to talk to him.

‘he’ – pronoun
‘himself’ – reflexive pronoun (anaphor)
Noun Phrases (NPs)

• Two types of NPs:
  • “Regular” Pronouns
    • he, she, them, yours, I, etc.

  • Reflexive Pronouns (Anaphors)
    • himself, herself, themselves, yourself, myself, etc.

• Pronouns and Anaphors – in complementary distribution (Chomsky 1981)
  • Across languages!
An Argument for/against Innateness

• Claim: The principles behind these phenomena – are innate.

• But, can the different properties of these NPs be learned?
  • By an RNN
NPs Representations in RNNs

• Training an RNN on a large language corpus

• Assessing how different are the representations of the model for pronouns and anaphors.

• Research strategies:
  • Clustering
  • Dimensionality Reduction
Objective: Translate input video frames into sentences.

https://www.youtube.com/watch?v=fFScL-7vWHM
CHALLENGES

• Translate data from one modality (video) to another (text).

• Different rate between video and text.

• Predict unseen words in evaluation.

• Learn the start and the end of a sentence.
Pre-trained CNN

CONV

LSTM

LSTM + Self Attention

Video feature extraction

Encoder

Feature Translation

LSTM + Attention

Pre trained on unsupervised language task.

Decoder

FC + Softmax

Output text

Input video
Multi Attribute Text Style Transfer

Or Yarnitzky & Or Patashnik
Style Transfer In Text - Example

Given a sentence:

frozen hot chocolate with peanut butter cups = amazing.
I’ll be back for some food next time!

An attribute of this sentence is sentiment = positive

We would like to “flip” the attribute value and generate new sentence:

frozen hot chocolate with peanut butter ? horrible. I’ll stick with the coffee shop next door!
Multi Attribute Text Style Transfer

• Flip more than one attribute value

• Example:
  • Input sentence
    exciting new show. john malkovich is superb as always. great supporting cast. hope it survives beyond season 1
    Attributes: sentiment = positive, category = movies
  • Output sentence
    nothing new. john grisham is not as good as his first book. not a good read.
    Attributes: sentiment = negative, category = books
Related Work

• Multiple-Attribute Text Style Transfer [Subramanian et al.] – ICLR 2019
  • Approach: DAE with back-translation technique

• Last year project – The Stylish Autoencoder
  • Approach: encoder-decoder setup with pretrained discriminator. Also uses the back-translation technique
Our Project

**Goal:** Take the approach suggested by The Stylish Autoencoder and adjust it to multi-attribute setting

**Ideas:**
1. Train encoder for each one of the attribute, and sum/concat them before giving them to the decoder
2. Train encoder-decoder for different attribute from the one in last year project, and apply it after that one
Tagging Patient Notes With ICD-9 Codes

Hadas Shaham
Tom Amsterdam
ICD-9/10 A Cornerstone Of Medical Care

• ICD-9 (International Statistical Classification of Diseases) is the international standard diagnostic tool for health management and clinical purposes

• Enables consistency among physicians in recording patient symptoms, diagnoses for clinical research and reimbursements claims

• the common terminology upon which most US health care payment systems are based and other major standards and practices have been built around it.
The Problem – Unstructured Text Classification

• multi-label classification problem - Assigning the appropriate ICD-9 code, describing the patient’s diagnosed medical status based on the unstructured text

• Extremely difficult task
  • ~13,000 labels
  • 20%-40% errors resulting in approximately 25B $ annual costs
The Project

• **Ideas**
  — Preprocess the dataset – fixing spelling mistakes and word replacement from relevant open sourced medical dictionaries
  — Embed the tokenized dataset using pre-trained word embeddings
  — Process the sentence while tagging important information (?)
  — Design and implement appropriate LSTM architecture
  — Enhance and improve the architecture using attention

• **Evaluate:**
  — F1 Score & AuC
  — State-Of-The-Art – ~0.7 F1 Score
NLP – Project
Fake News Detection

Or Wolkomir, Shachaf Ben-Gal
TAU University – Faculty of Computer Science
Fake News are Everywhere!

Top 10 Fake News Articles by Facebook Engagements

1. Lottery winner arrested for dumping $200,000 of manure on ex-'boss'’s lawn
2. Former first lady Barbara Bush dies at 92
3. Woman sues Samsung for $1.8M after cell phone gets stuck inside her vagina
4. BREAKING: Michael Jordan Resigns From The Board At Nike-Takes 'Air Jordans' With Him
5. Donald Trump Ends School Shootings By Banning Schools
6. Florida Man Arrested For Tranquilizing And Raping Alligators In Everglades
7. Two altar boys were arrested for putting weed in the censer-burner
8. North Korea Agrees To Open Its Doors To Christianity
9. Man Eats Girlfriends Booty For The First Time Time Dies From E. Coli
10. Muslim Figure: "We Must Have Pork-Free Menus Or We Will Leave U.S." How Would You Respond This?

Difficulty in detection, which rely on human reporting mechanism

Usually published in the purpose of misleading its readers for gaining damage, chaos and political or financial profits
Our Goal

- Automatically detect article as fake news for reducing its exposure with no human reporting it first.
Evaluated NN Models

**LSTM**
- Using pre-trained GloVe embeddings
- Bi-LSTM

**CNN**
- Using pre-trained GloVe embeddings

**HAN**
- Bidirectional GRU
- Attention for word-level and sentence-level encoding

**Convolutional HAN**
- One convolutional layer before Bi-LSTM

**C-LSTM**
- convolutional + LSTM layers

**Character-level C-LSTM**
- Character-level embedding
- convolutional + Word level LSTM + Bi-LSTM layers
Research Questions

◦ Expanding the evaluation of the different NN architectures using different types of embedding, such as sentence embedding, document embedding etc.

◦ Evaluating addition of non-textual based features where available, such as author, text length, words frequencies, users’ engagement etc.

◦ We will compare our results to the models presented in the article of Khan, Junaed Younus, et al (2019).
Predicting subjective aspects of question-answering
Kaggle competition by Google

Itay Manes & Dana Gueron
Computers are really good at answering questions with single answers.

Humans are still better at answering questions about opinions, recommendations, or personal experiences.

The goal is to improve automated understanding of complex question answer content.
Data description & evaluation

● Paired of questions and answers were gathered from nearly 70 different websites.

● $X$ - question_title, question_body, answer, category, users’ details

● Predict target values of all 30 labels for each question-answer pair.

Some examples:
  ○ question_fact_seeking
  ○ question_interestingness_others
  ○ answer_helpful

● Evaluation: mean label-wise of Spearman’s correlation coefficient
Our point of view

- Common-sense and subjective aspects are very complex matters that are very hard to define and to assess (but interesting!).

- Ideas:
  - Start with an EDA: exploring distributions of questions-answers scores, categories etc.
  - Embedding of words\characters.
  - Implementation of a contextualized models with variations (hyper-parameters, body\title and etc.)
  - Exploit the users-information
Fine-Grained Named Entity Recognition Using Adversarial Loss

Amit Attia and Noga Bar
The Task

- NER (coarse)

1. Bill robbed John, and he was arrested shortly afterwards.
2. Nvidia hands out Titan V for free to AI researchers.

- Ultra fine-grained named entity recognition
  - Any frequent nouns from dictionary is allowed as a type (10K vocabulary)

<table>
<thead>
<tr>
<th>Sentence with Target Entity</th>
<th>Entity Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>During the Inca Empire, {the Inti Raymi} was the most important of four ceremonies celebrated in Cusco.</td>
<td>event, festival, ritual, custom, ceremony, party, celebration</td>
</tr>
<tr>
<td>{They} have been asked to appear in court to face the charge.</td>
<td>person, accused, suspect, defendant</td>
</tr>
<tr>
<td>Ban praised Rwanda’s commitment to the UN and its role in {peacemaking operations}.</td>
<td>event, plan, mission, action</td>
</tr>
</tbody>
</table>
Limited Label Coverage

• Crowed sourced data with huge label space
• A sentence with target entity may miss some of the correct entity types

<table>
<thead>
<tr>
<th>Example</th>
<th>Bruguera said {he} had problems with his left leg and had grown tired early during the match.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Annotation</td>
<td>person, athlete, player, adult, male, contestant</td>
</tr>
<tr>
<td>Prediction</td>
<td>person, athlete, player, adult, male, contestant, defendant, man</td>
</tr>
</tbody>
</table>

• We want to punish our model less for reasonable mismatch between prediction and ground-truth
Adversarial Loss

• One target have multiple possible entities
  • Similarly to tasks like image coloring

• Adding an adversarial loss to the model predictions
  • Discriminate pairs of (input, predictions) and (input, ground-truth)
  • Challenge of differentiability
Codenames Solver

Roi Pomerantz, Roee Kattaby, Danielle Cohen
The Game

- Red team, Blue team
- 25 words on the board (Red, Blue, Black, White)
- Each team has a leader
- On each turn the leader provides a hint to his team and a number n.
- When a word gets picked it is eliminated from the board
- first team to eliminate all its word wins
- team that picks the black word loses.
Solution Description

- We will use 2 networks (GAN style) in order to solve the game:
  - A leader network will generate a “hint” word that relates to n words
  - A team network will predict which words should be picked
- Data: we can easily generate games, by sampling words and assign colors to the words.
- Gradients will propagate from the team network to the leader network when the team network makes mistakes.
Ensemble algorithm
Ensemble algorithm

Pre-tagged Corpus
## Ensemble algorithm

<table>
<thead>
<tr>
<th>Pre-tagged Corpus</th>
<th>f1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tagger Algo #1</td>
<td>70%</td>
</tr>
<tr>
<td>Tagger Algo #2</td>
<td>60%</td>
</tr>
<tr>
<td>Tagger Algo #3</td>
<td>35%</td>
</tr>
<tr>
<td>Tagger Algo #4</td>
<td>25%</td>
</tr>
</tbody>
</table>
Ensemble algorithm

Pre-tagged Corpus | f1
---|---
Tagger Algo #1 | 70%
Tagger Algo #2 | 60%
Tagger Algo #3 | 35%
Tagger Algo #4 | 25%
Ensemble algorithm

<table>
<thead>
<tr>
<th>Pre-tagged Corpus</th>
<th>f1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tagger Algo #1</td>
<td>70%</td>
</tr>
<tr>
<td>Tagger Algo #2</td>
<td>60%</td>
</tr>
<tr>
<td>Tagger Algo #3</td>
<td>35%</td>
</tr>
<tr>
<td>Tagger Algo #4</td>
<td>25%</td>
</tr>
</tbody>
</table>

New Tagger
Ensemble algorithm

<table>
<thead>
<tr>
<th>Pre-tagged Corpus</th>
<th>f1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tagger Algo #1</th>
<th>70%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tagger Algo #2</th>
<th>60%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tagger Algo #3</th>
<th>35%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tagger Algo #4</th>
<th>25%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>New Tagger</th>
<th></th>
</tr>
</thead>
</table>
Ensemble algorithm

<table>
<thead>
<tr>
<th>Pre-tagged Corpus</th>
<th>f1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tagger Algo #1</td>
<td>70%</td>
</tr>
<tr>
<td>Tagger Algo #2</td>
<td>60%</td>
</tr>
<tr>
<td>Tagger Algo #3</td>
<td>35%</td>
</tr>
<tr>
<td>Tagger Algo #4</td>
<td>25%</td>
</tr>
<tr>
<td>New Tagger</td>
<td></td>
</tr>
</tbody>
</table>
Ensemble algorithm

<table>
<thead>
<tr>
<th>Pre-tagged Corpus</th>
<th>f1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tagger Algo #1</td>
<td>70%</td>
</tr>
<tr>
<td>Tagger Algo #2</td>
<td>60%</td>
</tr>
<tr>
<td>Tagger Algo #3</td>
<td>35%</td>
</tr>
<tr>
<td>Tagger Algo #4</td>
<td>25%</td>
</tr>
<tr>
<td>New Tagger</td>
<td></td>
</tr>
</tbody>
</table>
Ensemble algorithm

<table>
<thead>
<tr>
<th>Tagger Algo</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>70%</td>
</tr>
<tr>
<td>#2</td>
<td>60%</td>
</tr>
<tr>
<td>#3</td>
<td>35%</td>
</tr>
<tr>
<td>#4</td>
<td>25%</td>
</tr>
</tbody>
</table>

New Tagger
Ensemble algorithm

Pre-tagged Corpus

Tagger Algo #1
70%

Tagger Algo #2
60%

Tagger Algo #3
35%

Tagger Algo #4
25%

New Tagger
Ensemble algorithm

<table>
<thead>
<tr>
<th>Tagger Algo</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-tagged Corpus</td>
<td>f1</td>
</tr>
<tr>
<td>Tagger Algo #1</td>
<td>70%</td>
</tr>
<tr>
<td>Tagger Algo #2</td>
<td>60%</td>
</tr>
<tr>
<td>Tagger Algo #3</td>
<td>35%</td>
</tr>
<tr>
<td>Tagger Algo #4</td>
<td>25%</td>
</tr>
<tr>
<td>New Tagger</td>
<td></td>
</tr>
<tr>
<td>Tagger Algo</td>
<td>Percentage</td>
</tr>
<tr>
<td>-------------</td>
<td>------------</td>
</tr>
<tr>
<td>Pre-tagged</td>
<td>70%</td>
</tr>
<tr>
<td>Corpus</td>
<td></td>
</tr>
<tr>
<td>Tagger Algo #1</td>
<td>70%</td>
</tr>
<tr>
<td>Tagger Algo #2</td>
<td>60%</td>
</tr>
<tr>
<td>Tagger Algo #3</td>
<td>35%</td>
</tr>
<tr>
<td>Tagger Algo #4</td>
<td>25%</td>
</tr>
<tr>
<td>New Tagger</td>
<td></td>
</tr>
</tbody>
</table>

Ensemble algorithm
Ensemble algorithm

<table>
<thead>
<tr>
<th>Pre-tagged Corpus</th>
<th>f1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tagger Algo #1</td>
<td>70%</td>
</tr>
<tr>
<td>Tagger Algo #2</td>
<td>60%</td>
</tr>
<tr>
<td>Tagger Algo #3</td>
<td>35%</td>
</tr>
<tr>
<td>Tagger Algo #4</td>
<td>25%</td>
</tr>
<tr>
<td>New Tagger</td>
<td></td>
</tr>
<tr>
<td>Pre-tagged Corpus</td>
<td>f1</td>
</tr>
<tr>
<td>-------------------</td>
<td>------</td>
</tr>
<tr>
<td>Tagger Algo #1</td>
<td>70%</td>
</tr>
<tr>
<td>Tagger Algo #2</td>
<td>60%</td>
</tr>
<tr>
<td>Tagger Algo #3</td>
<td>35%</td>
</tr>
<tr>
<td>Tagger Algo #4</td>
<td>25%</td>
</tr>
<tr>
<td>New Tagger</td>
<td>90%</td>
</tr>
</tbody>
</table>

Ensemble algorithm
For example:

Our New Tagger
For example:

“Average” sentence

Our New Tagger
For example:

“Average” sentence

Difficult grammar

Our New Tagger
For example:

- “Average” sentence
- Difficult grammar
- Heavy word morphology

Our New Tagger
For example:

“Average” sentence

Difficult grammar

Our New Tagger

Heavy word morphology

Very long sentences
Our New Tagger

For example:

“Average” sentence

Difficult grammar

Heavy word morphology

Very long sentences

Train a Classifier to refer sentences to their most accurate tagger
Using the comparison analysis as our data
For example:

“Average” sentence

Difficult grammar

Heavy word morphology

Very long sentences

Our New Tagger

Train a Classifier to refer sentences to their most accurate tagger
Using the comparison analysis as our data

Research question: Can we improve the f1 of the best algo

Measurement: comparing new classifier f1 with best algo f1
For example:

"Average" sentence

Difficult grammar

Heavy word morphology

Very long sentences

Our New Tagger

Train a Classifier to refer sentences to their most accurate tagger
Using the comparison analysis as our data

**Research question**: Can we improve the f1 of the best algo

**Measurement**: comparing new classifier f1 with best algo f1

**Plan B**, investigate the “dead zone”
Textual entailment

Tagging with explanations

VS

Tagging Without explanations

Liad Erez
Shahar Davidovich
Textual entailment task

Given a pair of sentences: premise and hypothesis, classify their relation as either:

- **Entailment**: if the premise entails the hypothesis
- **Contradiction**: if the hypothesis contradicts the premise
- **Neutral**: if neither entailment nor contradiction hold.

Applications:
- Question answering
- Information extraction
- Summarization
The Dataset: SNLI (Stanford natural language inference)

We aim to train our models on the SNLI dataset which contains 570K data points of human-generated triples (premise, hypothesis, label)

Example data point:

Premise: An adult dressed in black holds a stick.
Hypothesis: An adult is walking away, empty-handed.
Label: contradiction
Camburu et al. extended the SNLI by adding natural language explanations to the relations between each premise-hypothesis pair.

Example Data point:
Premise: An adult dressed in black holds a stick.
Hypothesis: An adult is walking away, empty-handed.
Label: contradiction
Explanation: Holds a stick implies using hands so it is not empty-handed.
The MultiNLI dataset

Contains 433K sentence pairs annotated with textual entailment information.

We will use this dataset for evaluation.
Our goal

We aim to train two independent models:

- A model that is trained on the SNLI dataset to produce labels.
- A model that is trained on the e-SNLI dataset to produce labels as well as explanations.

We aim to check if the second model generalizes to the MultiNLI dataset better than the first model.

If this is indeed the case, it may suggest that the second model contains more information that is useful for understanding natural language than the first model.

Such a conclusion can also be relevant to other NLP tasks other than textual entailment.
Thanks!
TEXT TO SQL – THE SPIDER CHALLENGE

Liat Bezalel
Given a natural language question we would like to translate it to a SQL query:

```
SELECT T1.country_name 
FROM countries AS T1 JOIN continents AS T2 ON T1.continent = T2.cont_id 
JOIN car_makers AS T3 ON T1.country_id = T3.country 
WHERE T2.continent = 'Europe' 
GROUP BY T1.country_name 
HAVING COUNT(*) >= 3
```
WHAT IS SPIDER?

- a large-scale complex and cross-domain semantic parsing and text-to-SQL dataset
- consists of 10,181 questions and 5,693 unique complex SQL queries on 200 databases with multiple tables covering 138 different domains.
- Different complex SQL queries and databases appear in train and test sets
Annotators check database schema (e.g., database: college)

Table name

Table 1  **instructor**  
| id | name | department_id | salary | .... |

Table 2  **department**  
| id | name | building | budget | .... |

Annotators create:

**Complex question**: What are the name and budget of the departments with average instructor salary greater than the overall average?

**Complex SQL**

```sql
SELECT T2.name, T2.budget
FROM instructor as T1 JOIN department as T2 ON T1.department_id = T2.id
GROUP BY T1.department_id
HAVING avg(T1.salary) >
(SELECT avg(salary) FROM instructor)
```
RELATED WORK

- IRNET
- GNN
- And more.. The best accuracy achieved by RATSQL v2 + BERT with 61.9 on the test set
IDEA 1: USE GLOBAL-GNN + BERT
IDEA 2: USE GLOBAL-GNN TO DECODE SEMQL
IMPROVING CONTEXTUALIZED EMBEDDING WITH GRAPH NEURAL NETWORKS

NLP Fall 2019
By: Danielle Hausler & Uri Katz
Motivation

- Syntax, relationships between words in a sentence, can help in many speech and natural language processing applications.

- Syntax-based embeddings encode functional similarity (in-place substitutable words) rather than topical similarity.

- Graphs are a kind of data structure which models a set of objects (nodes) and their relationships (edges).

- GNN is capable of providing rich local contextual and structural information by encoding edge or node attribute features.

- Graph structures are common in many NLP tasks.
Mission

- Combine the power of pre-trained BERT which generates contextualized embedding with syntactic information.

- We will evaluate our new representations on different Intrinsic Tasks such as Word Similarity, Concept Categorization, Word Analogy
Character to word embeddings adapter

Yannay Jaffe and Lior Ben-Moshe
Motivations

- Pre-trained word embeddings can be large (~1-2GB for GloVe).
- We might encounter new words during test time – we might not have an embedding for those words.
- Pre-trained character embeddings are useful in many cases and achieve state of the art performance in several NLP tasks (sometimes in conjunction with contextualized word embeddings).
- Some models have an “API” accepting word embeddings. Can use an arbitrary character to word embeddings adapter and learn along the way, but it might be useful to use pre-trained adapter.
Idea

- Define a self-supervised learning task for generating word embeddings from pre-trained character embeddings.
- After the training, can use the trained character to word embeddings adapter in word embeddings-based NLP models (and adjust along the way).
- Test the generated embeddings in some real NLP tasks such as sentiment analysis or tagging and compare to “regular” word embeddings.
A possible model and learning task
Plug and Play Language Models
Text Generation with a Sophisticated Discriminator

Anna Vieweg-Konrad
Language Generation

- Today: good results in text generation capabilities trained on large datasets
- **But** attribute control requires
  - modify model architecture
  - fine-tune on attribute specific data (retrain)
- **Solution**: A Plug and Play Language Models (PPLM) - a pre-trained language model (LM) combined with a simple attribute classifiers

Attributes
- sentiment: negative, positive, neutral, ...
- topic: science, politics, ...
- toxicity
- ...
PPLM - a Simple Idea

- Let \( x \) be a generated sample text and \( a \) the desired attribute

<table>
<thead>
<tr>
<th>Model type</th>
<th>Form of model</th>
<th>Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language Model</td>
<td>( p(x) )</td>
<td>Uncond.</td>
</tr>
<tr>
<td>Fine-tuned Language Model</td>
<td>( p(x) )</td>
<td>Uncond.</td>
</tr>
<tr>
<td>Conditional Language Model</td>
<td>( p(x</td>
<td>a) )</td>
</tr>
</tbody>
</table>

- Want to model \( p(x|a) \) without changing the sample data
- Use base generative model \( p(x) \) (pre-trained, unconditioned model, e.g. Transformer, RNN)
- **Solution** \(^1\): Plug in an attribute model

\[
p(x|a) \propto \underbrace{p(a|x)} p(x)
\]

\(^1\)by Dathathri et al. 2019
Conditional Attribute Model

Two approaches to model $p(a|x)$ \(^2\):

- **Topic classifier**: Bag of words (BoW).
- **Sentiment classifier**: *Linear discriminator* trained on top of the pre-trained generative language model latent representations.

Output example trained on the SST-5 dataset:

<table>
<thead>
<tr>
<th>Type</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>[-]</td>
<td>The potato and cauliflower are both in season to make combo breads, mounds, or pads. For an added challenge, try some garlic mashed potatoes.</td>
</tr>
<tr>
<td>[Negative]</td>
<td>The potato is a pretty bad idea. It can make you fat, it can cause you to have a terrible immune system, and it can even kill you...</td>
</tr>
<tr>
<td>[Positive]</td>
<td>The potato chip recipe you asked for! We love making these, and I’ve been doing so for years. I’ve always had a hard time keeping a recipe secret. I think it’s the way our kids love to eat them – so many little ones.</td>
</tr>
</tbody>
</table>

\(^2\)by Dathathri et al. 2019
Proposal

Linear Discriminator
- Deploy a small neural network (NN) instead of a linear discriminator within the Transformer infrastructure to control for sentiment
  - Theoretical analysis of the optimization problem
  - Train on the SST-5 dataset and compare to previous results

Dataset (with original linear discriminator)
- Reuters-21578 dataset: single labelled news articles by topic as appeared on the Reuters newswire in 1987
- Cornell dataset: attributes subjectivity vs. objectivity
Cycle Loss for Unsupervised Translation

Lior
Gal Cohen
Goal

• Given any sentence classification task and corresponding datasets of two different languages - train translation models between the two languages.

• The training is unsupervised

• Loss: Reconstruction and Classification.

• Evaluation: All translation metrics. Competing against other unsupervised methods
Obama was born on _____
Fine Tuning
When was Obama born?
When was Obama born?

Obama was born on ______
1. How much do we get from pre-training?
2. How hard it is for the model to change format?
3. Can this help us with transfer learning?
4. Can this help us to enlarge certain training sets?
5. Can we leverage this method to other tasks?