Alternative Word Embeddings

Alexander Owens, Saksham Goyal

Introduction

- Text representations in LLM
  - Tokenization
  - Text compression (GZIP)
- Is information density a useful predictor?
  - Compression length
  - Kolmogorov distance
- Possible use cases
  - Fast runtimes on abnormal datasets
    - Lack of pretrained models
  - Sequence conversion

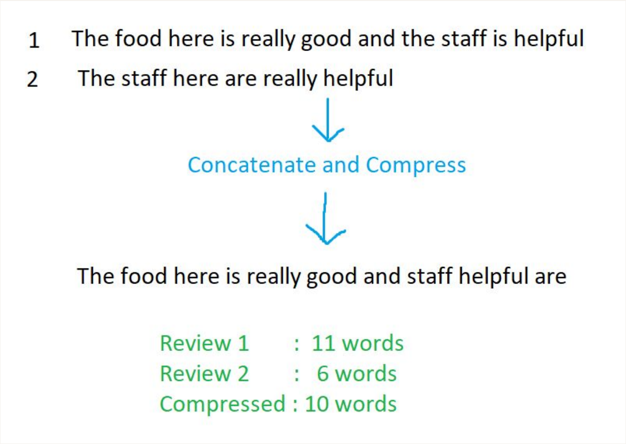
Problem Statement

- We want to create a model that can classify Yelp reviews as positive, negative, or neutral
- Most solutions tend toward using a neural network, or large complex algorithms.
- Create a model that uses as little resources as possible while maintaining high accuracy

Solution:

- Correlate compression length with sentiment using GZIP
- Use a KNN model to classify text

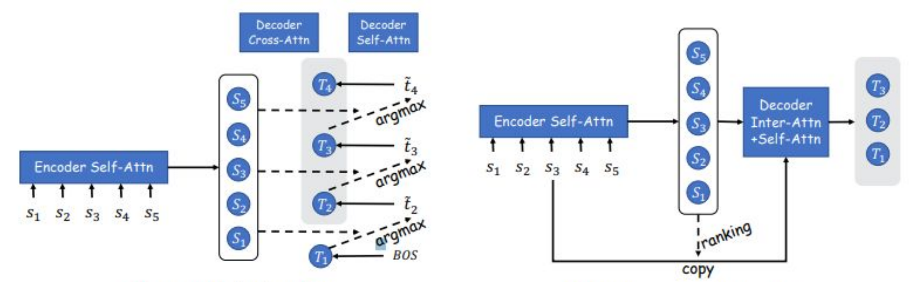
KNN Distance Metric



Related Work

Text Compression-Aided Transformer Encoding

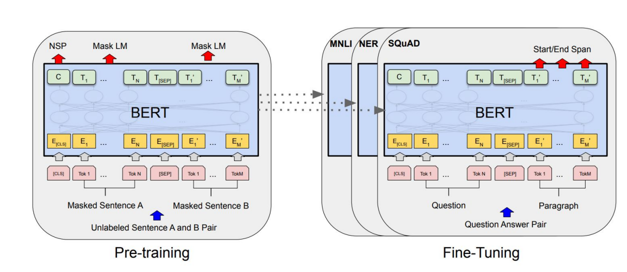
- Compares implicit vs explicit text compression
- Concludes text compression algorithms improve over implicit text compression
- Trained on general cases:
  - Machine reading
  - Machine translation
- Differentiates between encoder and decoder sided text compression integration



(a) Autoregressive decoding (b) Non-autoregressive decoding

BERT

- An encoder model that uses transformers
- It can understand context from forwards and backwards which allows it much greater understanding of the sequence of text
- It can do a wide variety of tasks from text classification to question answering. It’s very powerful



Data Exploration

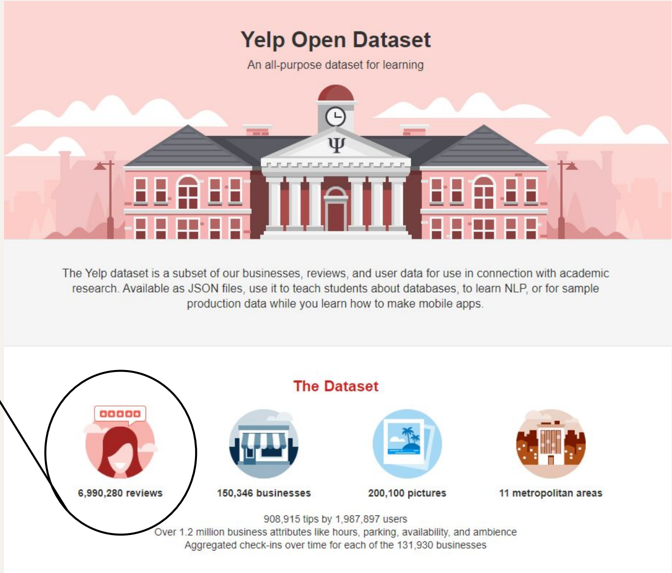
Yelp Open Dataset

- Interested in the Reviews section from the dataset
- Dataset is in JSON format
- We needed 3 columns
  - Review Text
  - Stars
  - Usefulness

review.json

Contains full review text data including the user_id that wrote the review and the business_id the review is written for.

```
{
    // string, 22 character unique review id
    "review_id": "zdSx_SD6obEhz9VrW9uAWA",

    // string, 22 character unique user id, maps to the user in user.json
    "user_id": "Ha3iJu77CxlrFm-vQRs_8g",

    // string, 22 character business id, maps to business in business.json
    "business_id": "tnhfDv5Il8EaGSXZGiuQGg",

    // integer, star rating
    "stars": 4,

    // string, date formatted YYYY-MM-DD
    "date": "2016-03-09",

    // string, the review itself
    "text": "Great place to hang out after work: the prices are decent, and

    // integer, number of useful votes received
    "useful": 0,

    // integer, number of funny votes received
    "funny": 0,

    // integer, number of cool votes received
    "cool": 0
}
```



Methodology

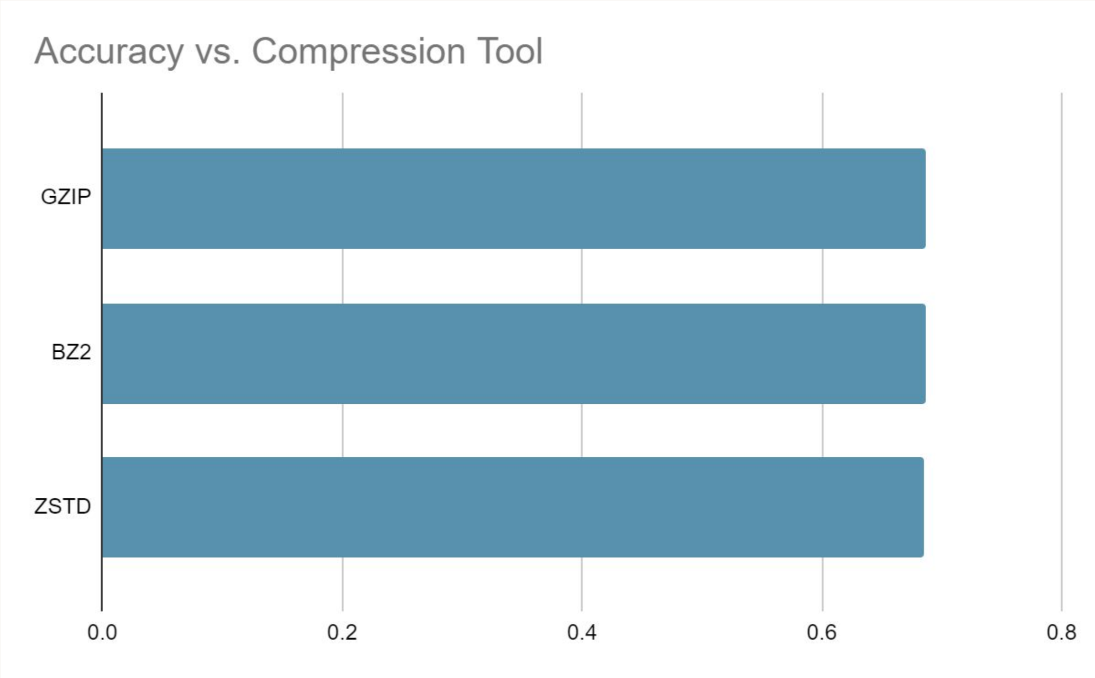
Dataset Preprocessing

```python
# Read the first 1 million rows
df = pd.read_json(f'{file}.json', lines=True, nrows=1_000_000)

# Remove all rows where the text has less than 100 characters to ensure high quality reviews
df = df[df['text'].str.len() > 100]

# remove unnecessary columns to save space and time
df.drop(['review_id', 'date', 'user_id', 'business_id'], axis=1, inplace=True)

# map [0,5] stars to negative (0), neutral (1), positive (2)
df['sentiment'] = df['stars'].map({0 : 0, 1 : 0, 2 : 0, 3 : 1, 4 : 2, 5 : 2})

# Remove all newlines and carriage returns from the text
df.replace('\n', ' ', regex=True, inplace=True)
df.replace('\r', ' ', regex=True, inplace=True)
df.replace('  ', ' ', regex=True, inplace=True)

# Duplicate rows based on the 'useful' voted reviews to bias the model towards helpful reviews
useful_duplicated = pd.DataFrame(df.reindex(df.index.repeat(df['useful'] + 1)).reset_index(drop=True))
useful_duplicated.to_csv(f'{file}.csv', header=True, index=False, mode='w')
useful_duplicated.head()
```

Compression Techniques

- Text Lossless compression
  - Keeps compression size similar between techniques
  - GZIP
    - Faster compression speed
    - Lower compression ratio
  - ZSTD
    - Slower compression speed
    - Higher compression ratio
  - BZ2
    - Median compression speed
    - Median compression ratio

Testing Text Compression Tools



Speed and Efficiency

- Given the ‘Useful’ column, marked when users thought a review was useful
  - Duplicated this row for every time that users thought the review was useful
  - Increases probability this row is picked when sampling for KNN
- When finding KNN, 10% of the dataset is used
  - Reduces prediction time, but reduces accuracy
- K value
  - SQRT(n/10), or the SQRT of the subsample size

Results

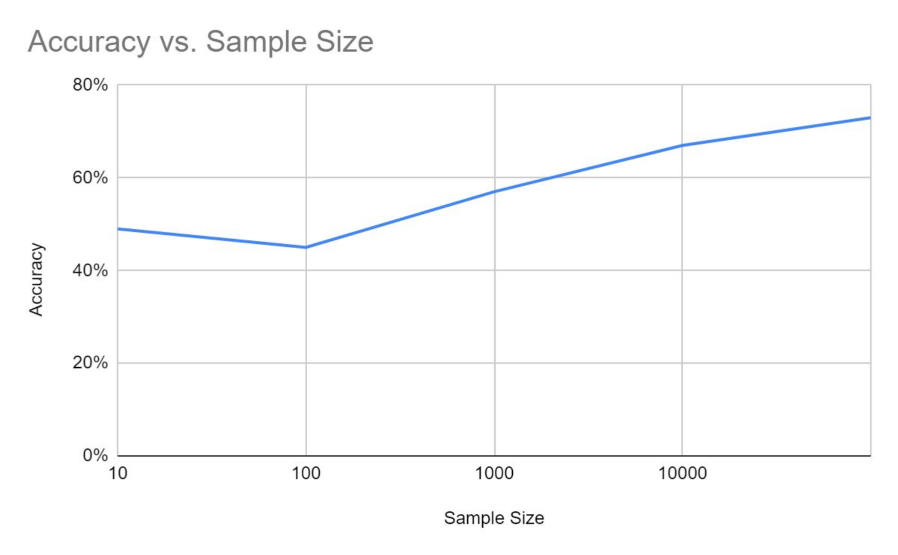
Performance

- 100,000 samples for the KNN with 10% sampling rate
- 20% test-train split
- Final Accuracy of 73%

```
print("Final Accuracy = ", accuracy_score(sample_test_labels, labels.reshape(-1)))
```

```
[10]
Fetching data...
number of samples loaded 100000
Fitting model...
(20000,)
Generating predictions...
Compressing input...: 100%|██████████| 20000/20000 [3:12:50<00:00,  1.73it/s]
Final Accuracy =  0.7308
```

Performance



Analysis

Conclusion

- We successfully were able to create the KNN model to classify Yelp reviews
- We achieved a final accuracy of 73%
    - Random guessing accuracy is 33% (3 classes)
    - Much better than random
- Worse compression ratios don’t necessarily mean worse accuracy
- Compression distance can be correlated with sentiment

How to make this better

- Use a higher sampling rate to improve the effectiveness of the KNN model
  - This will be much slower, but will increase the accuracy
- Instead of compressing the raw text consider using the word embeddings of the reviews instead.
  - Word embeddings hold much more information about the context of the words

Future Work

- Worse compression ratios don't necessarily mean worse accuracy
  - Try again with a much faster lossless compression algorithm

- Try to include Kolmogorov distance alongside a transformer as a predictor

- Use binary sequence generated compression tool as input to a transformer

- Try adding extra information into the compression (likes, reactions, etc.)

Work Cited

- https://aclanthology.org/2023.findings-acl.426.pdf
- https://github.com/bazingagin/npc_gzip
- https://yelp.com/dataset
- https://youtube.com/watch?v=jkdWzvMOPuo
- https://ieeexplore.ieee.org/abstract/document/9354025