Movie Review Sentiment Predictor

Colin Johnson

DS5110

Introduction

- The goal of the project is to make a model that can predict the sentiment score of movie reviews
- Use ML methods of previous movie reviews in order to build a model that will use prior data give better predictions.
- From there be able to sort the data into groups of positive or negative.

Methodology

- Used simple preprocessing task to remove html tags, remove punctuation, remove irrelevant words, and more.
- Padded reviews as potential loss of information at the end of the reviews. The model only works if all inputs (reviews) are of the same length. Since we set the set the tokens to an ID, we pad with 0s. Ex: (["I love the movie")] -> (['love']) -> ([200])

```
nlp = spacy.load('en_core_web_sm')
def remove_html_tags(text):
    return re.sub(r"<.*?>", "", text)
def spacy_lemmatization(text):
    doc = nlp(text)
    return [token.lemma_ for token in doc]
def filter_vocab(vocab, words_remove):
    #filter, function applied in tokenizing the vocab
    return {word for word in vocab if word not in words_remove}
def tokenize_with_vocab(text, vocab):
    '''
    Tokenizing the reviews, removing common and useless words, etc.
    '''
    words_remove = ['movie', 'film', 'director', 'plays', 'horror', 'comedy', 'watching', 'seen', 'people', 'guy', 's', 'time', 'second', 'boo
    vocab = filter_vocab(vocab, words_remove)
    tokenized_reviews = []
    for doc in nlp.pipe(text, batch_size=1000, disable=["parser", "ner", "tagger"]):
        cleaned_text = remove_html_tags(doc.text)
        lemmas = spacy_lemmatization(cleaned_text)
        tokens = [
            lemma.lower() for lemma in lemmas
            if lemma.lower() in vocab and lemma.lower() not in nlp.Defaults.stop_words
        ]
        tokenized_reviews.append(tokens)

    return tokenized_reviews
def tokenize_id(tokens, vocab):
    '''
    Retrieving the id values of the tokens corresponding with the vocab dicitionary
    '''
    return [vocab[token] for token in tokens]
def pad_sequences(sequences, max_length):
    '''
    Padding data so each review length is equal, necessary for the model
    '''
    return rnn_utils.pad_sequence(sequences, batch_first=True, padding_value=0)[:, :max_length]
```

Methodology 2

- Used a CNN model to look at each padded token. The model would extract features from input values and connect it with the label. Ex: [a'love'] -> [200] would be connected with a positive sentiment score and from there the model can use this knowledge to be able to predict scores with other values as well.

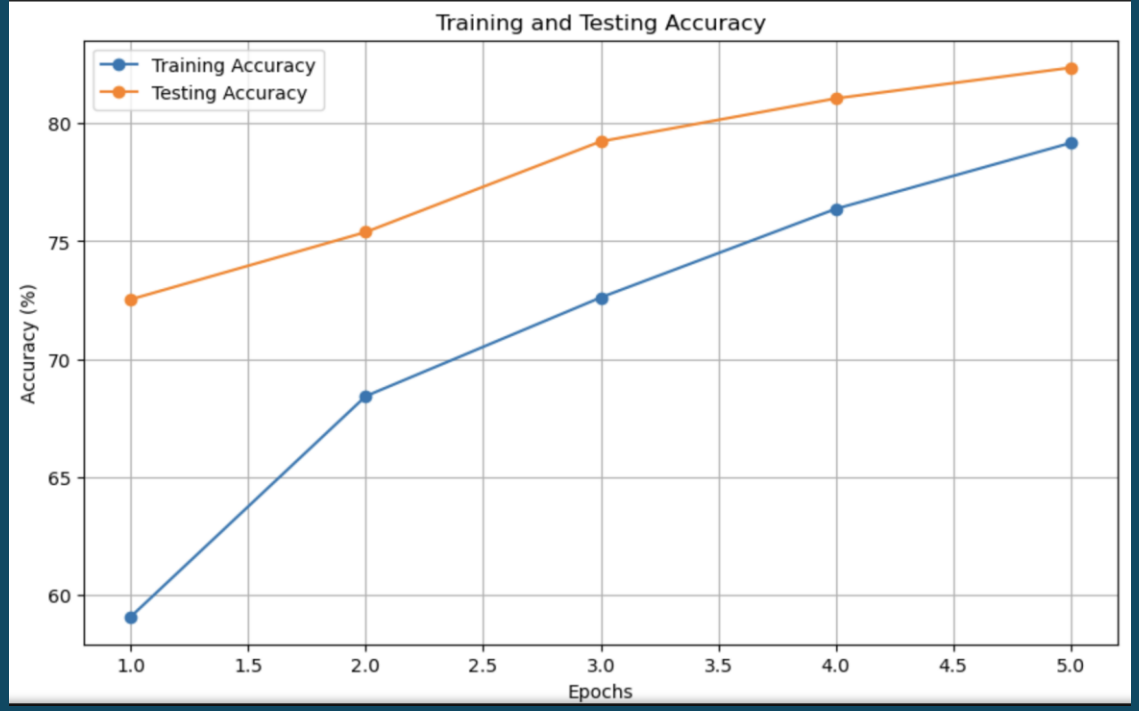
```
class TextCNN(nn.Module):
    def __init__(self, vocab_size, embedding_dim, conv_config, output_size, dropout=0.5):
        super(TextCNN, self).__init__()


        self.conv_config = conv_config
        self.output_size = output_size
        self.dropout_p = dropout

        self.embedding = nn.Embedding(vocab_size, embedding_dim)
        #configuring the hidden layers, extracting features from the input nodes (reviews)
        self.convolutions = nn.ModuleList([
            nn.Sequential(
                nn.Conv1d(embedding_dim, self.conv_config['num_channels'], kernel_size=kernel)
                nn.ReLU(),
                nn.AdaptiveMaxPool1d(1)
            )
            for kernel in self.conv_config['kernel_sizes']
        ])
```

WHY CNN? "Convolutional Neural Networks (CNNs) are used for text classification because they excel at identifying important patterns and features within text data, regardless of their position in the sentence, making them particularly effective for tasks where the local context of words matters, like sentiment analysis or topic classification, where key phrases can appear anywhere in the text."

Methodology 3/(Partial) Results

- Finally, we test the strength of our model by running an evaluation.
- Used 5 epochs, found 80% accuracy, expect more with 10.
- Used criterion to collect test & loss values to test strength

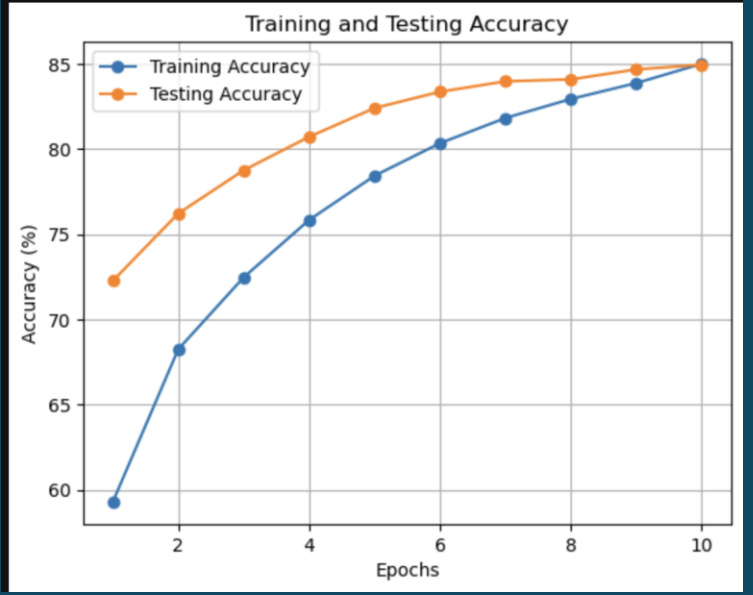
```
Epoch 5/5:
Train Loss: 0.4554, Train Acc: 79.17%
Test Loss: 0.3981, Test Acc: 82.36%
Best Test Acc: 82.36%
----------------------------------------------------------------
```



Results

- Ran with 10 epochs found 85% accuracy.
- Ran test cases for program to check sentiment score of reviews generated by AI.

```
Epoch 10/10:
Train Loss: 0.3501, Train Acc: 85.01%
Test Loss: 0.3465, Test Acc: 84.95%
Best Test Acc: 84.95%
----------------------------------------------------------------
```




```
Review: This movie is a gem that reminds us why we love cinema. From the very first scene, it pulls you into a vivid world of compelling char
acters, breathtaking visuals, and a narrative that keeps you emotionally invested until the very last frame. The performances are nothing sho
rt of extraordinary. The lead actor delivers a career-defining role, capturing every nuance of the character's journey. The supporting cast i
s equally remarkable, creating a dynamic ensemble that feels authentic and relatable. The director's vision is evident in every scene, blendi
ng artistry and storytelling in a way that feels seamless. The cinematography is stunning, with each frame looking like a carefully crafted p
ainting. The music score adds another layer of depth, perfectly complementing the emotional highs and lows of the story. What truly sets this
movie apart is its heart. It's a film that resonates on a deeply personal level, leaving you with a sense of hope, wonder, and inspiration. W
hether you're a fan of heartfelt dramas, epic romances, or thought-provoking narratives, this film has something for everyone. A must-watch f
or anyone who loves great storytelling, this movie will stay with you long after the credits roll.
Predicted Sentiment: Positive Review

Review: This movie is a huge disappointment that fails to live up to its promises. From the very first scene, it struggles to capture your at
tention and quickly becomes tedious. The characters are bland and uninteresting, with performances that lack depth. The plot is predictable a
nd meanders aimlessly, never building any real tension or excitement. The director's vision is absent, and the cinematography feels lazy and
uninspired. The music score is generic and adds nothing to the story. Overall, it's a film that fails to connect on any meaningful level, lea
ving you feeling empty and unsatisfied. A total miss for anyone expecting a compelling cinematic experience.
Predicted Sentiment: Negative Review
```

Limitations

- Found with smaller reviews (phrases or reviews less than two sentences the accuracy is much lower at around 60%.

- Longer reviews found around 99% accuracy
- This causes very short reviews to produce many false positives.
- Solution is more training data for the model. This means higher accuracy (take more from other datasets)

```
"This is a beautifully written movie!",
"This movie is an absolute masterpiece, delivering a cinematic experience that is both deeply moving and visually stunning. From the very firs
The director's vision is evident in every frame, crafting a story that is as thought-provoking as it is emotionally resonant. The cinematograp
What sets this film apart is its ability to balance grand, epic storytelling with intimate, personal moments that strike a universal chord. It
```

```
Review 4: Positive Review (Confidence: 0.55)
Review 5: Positive Review (Confidence: 0.99)
```

Conclusion

- As addressed earlier shorter reviews have much lower accuracy so in the future address this would make the model more accurate. (Will increase test accuracy greatly too)
- With more time make a small HTML webpage to demo if reviews are positive or not then implement for review websites where it flags if a review is positive or not.
- Testing the model with more diverse datasets as well will increase overall accuracy before implementing this however