CONTENTS

Cloudera Fast Forward

Textflix: Using Transfer Learning for a Natural Language Processing Prototype

The Textflix prototype

Perhaps the most exciting aspect of transfer learning is that it is so broadly applicable. It does not

enable one specific capability, like question

CONTENTS

answering, language translation, text classification, or search. It makes *each* of these more accurate, at a lower cost, with less data, and thus more accessible to a broader community. It opens the pathway from research to production.

We built a prototype, Textflix, that leverages transfer learning for sentiment analysis. Textflix performs positive or negative sentiment detection on movie reviews, which present a challenge to text processing systems because of the complex ways in which humans express their preferences. We added LIME, an interpretability technique, to provide insight into the

model's predictions and used this mechanism to provide individual summaries of each movie. The entire product was built using a model trained on just 500 labeled examples. The modeling

☰ CONTENTS

was implemented with off-the-shelf tools from AllenNLP and did not require writing any code. Everything was completed within an infrastructure budget of $25.

Sentiment

While transfer learning is likely to be beneficial for almost any NLP application, some will benefit more than others. Even though transfer learning can be implemented at low cost, we do not recommend using a sophisticated deep learning model for a 1% gain in accuracy. Some applications, like basic

topic categorization, can be handled admirably by extremely simple statistical models (like Naive Bayes, which is essentially counting words). In such cases, sticking with the simple model works best.

But sentiment detection – the task of inferring how

☰ CONTENTS

the writer or speaker feels – requires sophisticated tools. When humans convey their feelings, they may use sarcasm and irony. They might incorporate obscure pop culture references. They

might have mixed feelings, contradict themselves, or even change their minds by the time they've finished their thoughts! Without transfer learning, all of these difficult challenges would need to be solved by learning from a single, possibly small, dataset. Because of these difficulties, sentiment detection demonstrates very clearly the power of transfer learning.

Dataset

Textflix is built on the IMDB dataset, a popular open dataset commonly used for sentiment analysis. The dataset consists of 25,000 movie reviews from the users of the popular online movie database, although only

☰ CONTENTS

500 of those were used to build the model behind Textflix, as a demonstration of the power of transfer learning.

While sentiment analysis can be applied in many different domains, movie reviews are interesting because they present a diverse set of challenges. Many of the reviews are straightforward and simple, but some contain subtle clues as to the author's opinion. These subtleties are challenging for machine learning models – a model based on simple statistics will not work well. Because the reviews are written in rather plain English (no specialized dialects or slang), publicly available pretrained models (which were trained on generic English) will work well.

Models

The modeling process for Textflix was extremely simple. One of the great

☰ CONTENTS

simpler. One of the great benefits of transfer learning is that it eliminates the need to invent complex new neural network architectures that uniquely solve a particular problem. Off-the-shelf transfer learning models already provide state-of-the-art accuracy, so the fewer changes we make, the better. After comparing several models, we ended up using the large version of the BERT model...

We experimented with several popular transfer learning models, and also compared their performance to simple but strong baseline methods. Although we built Textflix with a model trained on only 500 examples, for each model we explored its performance curve when training on more labeled examples. In the following sections we present these results and justify our modeling choices.

≡ **CONTENTS**

Baseline Models

Testing baseline methods is an important first step in the modeling process. Even if they are unlikely to yield a usable model, they

are easy to implement, present a logical reference for comparing future models to, and are usually more interpretable than more sophisticated alternatives. In some cases, the simplicity and interpretability advantages they present may outweigh the decrease in accuracy in the results they produce. For our prototype, we explored two baselines: SVM with Naive Bayes features and word vectors.

NB-SVM

For text classification problems like sentiment analysis it makes sense to choose a simple model based on bag-of-words as the first baseline. In many text classification

≡ **CONTENTS**

problems, like topic classification, these types of baseline models may even be the best choice. NB-SVM treats the text as a bag of words and combines a Naive Bayes model (also a reasonable baseline) with a support vector machine. This model has been shown to produce strong linear baselines for text classification, and sentiment analysis in particular.
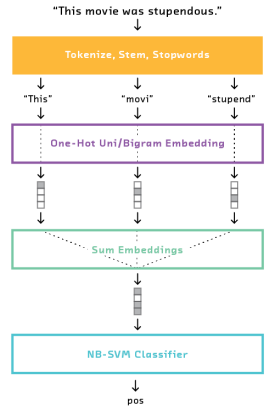
Pipeline architecture for the NB-SVM model.

In testing this model we generated both uni- and bigram features for the

☰ CONTENTS

NB-SVM classifier, removed stopwords from the input, and used a Snowball stemmer to normalize each word.

The performance of the NB-SVM was poor and unpredictable at low dataset sizes – it simply did not have enough observations to learn which words strongly correlated with positive or negative sentiment. However, at larger training set sizes (e.g., 10,000 examples), this baseline reached a useful accuracy

of about 85% with very little tuning.
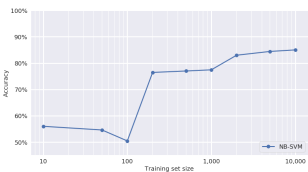
NB-SVM is no better than guessing at training set sizes less than 200.

This performance curve shows that even simple bag-of-words models can identify a large majority of sentiment examples

≡ CONTENTS

correctly. And because the NB-SVM model is simple and fast to implement, it would be easier to support in a production use case than some deep neural network models.

Word Vectors

As we've seen, the NB-SVM model cannot be

reasonably expected to perform well with small training datasets. Because it is a bag-of-words model, it has no a priori knowledge of words and their associations and must learn everything from the training data. A model that leverages word vectors should do better in the small-data regime, since word vectors are a form of transfer learning. That is, the meaning of words is already captured in the pretrained word vectors, and is not affected by the small data size.

We used the following simple architecture for

this model.

CONTENTS
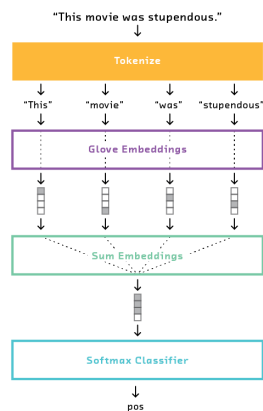
```
Pipeline architecture for
the word vectors model.
```

The performance curve shows that the word vectors model generally

follows the same trend as the NB-SVM baseline. (It is important to note that there is a lot of variance at each data point, which is not shown in the plot, so small differences should be taken lightly.)

```
Transferred word vectors
suffer from the same
problems as the NB-SVM
model.
```
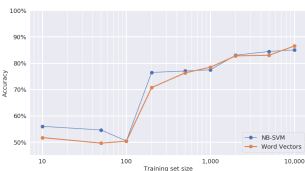
This result is, in some ways, surprising since word vectors are a form of transfer learning and should therefore be more resilient to limited data than the baseline NB-SVM model. It is likely that the word vectors model could be tuned to outperform NB-SVM at smaller training sizes by adjusting the hyperparameters and architecture. However, we purposely spent little time optimizing hyperparameters with any of the models. The

hyperparameter optimization step is largely a heuristic process and can require deep expertise to guide the search – something we wanted to avoid.

Transfer Learning Models

With well-performing

☰ CONTENTS

baselines established, we began to try out several of the publicly available transfer learning models. We experimented with ULMFiT using the Fast.AI library and found it to perform well, even with limited data. We performed language-model fine-tuning using 50,000 unlabeled examples for the ULMFiT model, and then performed supervised training at various dataset sizes. In addition to ULMFiT, we ran experiments for both the BERT-Base and BERT-Large models, using the AllenNLP library.
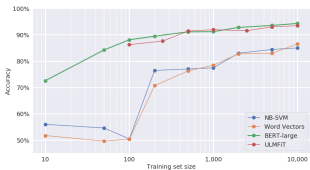
New transfer learning models can perform well even with fewer than 100 examples.

The best-performing models were the BERT-Large and ULMFiT models. While these models

☰ CONTENTS

produced nearly equal results, we found that the BERT-Large model was easier to implement and

experiment with. This is in part because no language model fine-tuning step was required with BERT-Large (target task fine-tuning, however, was performed for each model), and in part because ULMFiT is trained via the Fast.AI library, which imposes development via notebooks – something we find to be restrictive.

We trained the BERT-Large model without making any custom modifications and defined the entire experiment in a JSON configuration file in the AllenNLP style. We trained on a single GPU for 20 epochs, using an Adam optimizer, and used gradual unfreezing for fine-tuning the layers of the model.

"This movie was stupendous."
↓

☰ CONTENTS
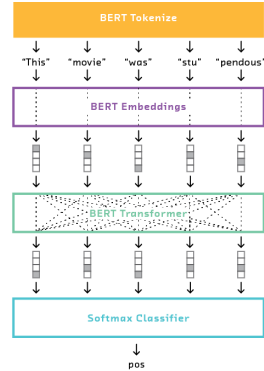
Pipeline architecture for the BERT-large model

Overall, we did very little tuning of the model's hyperparameters. This limited tuning requirement is one of the greatest benefits of transfer learning: the out-of-the-box performance is already very good, and eking out a further 1-2% of accuracy has diminishing returns and would require expensive-to-acquire knowledge of the model architecture.

The final BERT model provides accuracy roughly equivalent to the state-of-the-art model, using just 500 labeled examples for training.

CONTENTS

Interpretability

A cost of using transfer learning is the less interpretable, black box nature of neural network models. The ability to detect sentiment is undeniably useful, but the

ability to explain those predictions is significantly more powerful. With interpretability, we can not only classify the user's sentiment but point to specific evidence to support that classification. We found the addition of interpretability to the model's predictions to be surprisingly useful. Here we will discuss our approach and some of the benefits.

classification: positive · 82.3% certainty
by consequential_countdown · 143 days ago
  So, you wanna be a rock star? See this movie. You don't like rock, you say? Or you're REALLY into heavy metal? Then put on your favorite album and dream yourself away, this movie has nothing to offer. Rarely have I ever seen a movie being able to portrait the dream of being in a rock band as good as this. I had long hair during the late 1980's and early nineties, and I have played guitar for the last 15 years or so. Did I like Rock Star? Oh yes. The music is good, not great, the actors are good, and believable, even Jennifer Aniston plays her part to perfection. And Mark Wahlberg is perfect as the wannabe rock singer. So you know what you're going to get. A movie about dreams coming true, being stepped on, and finally figuring out what life is really about. It's a good solid seven out of ten, no more, no less.

classification: positive · 82.3% certainty
by consequential_countdown · 143 days ago
  So, you wanna be a rock star? See this movie. You don't like rock, you say? Or you're REALLY into heavy metal? Then put on your favorite album and dream yourself away, this movie has nothing to offer. Rarely have I ever seen a movie being able to portrait the dream of being in a rock band as good as this. I had long hair during the late 1980's and early nineties, and I have played guitar for the last 15 years or so. Did I like Rock Star? Oh yes. The music is good, not great, the actors are good, and believable, even Jennifer Aniston plays her part to perfection. And Mark Wahlberg is perfect as the wannabe rock singer. So you know what you're going to get. A movie about dreams coming true, being stepped on, and finally figuring out what life is really about. It's a good solid seven out of ten, no more, no less.

```
Predicting sentiment is
```

CONTENTS

```
useful, but pointing
readers to specific
evidence that the model
relies on allows the
reader to trust the
prediction.
```

Modified LIME for Sequences

We used the technique of Local Interpretable Model-agnostic Explanations (LIME) to add interpretability. LIME can be applied to almost any type of model, but here we will consider its application to text data.

Put simply, LIME is a way to understand how different parts of an input affect the output of a model. This is accomplished, essentially, by turning the dials of the input and observing the effect on the output.

Consider a model that classifies news headlines as "clickbait" or "not clickbait." The model could identify individual words as more or less clickbaity, depending on

CONTENTS

how they affect the output. If removing a word from the input generally makes the model predict clickbait, then that word contributes to the model's clickbait classification for that particular input.
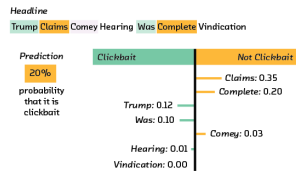
LIME identifies words that contribute to positive or negative classification.

This method makes sense for models that process the input as a bag of words, but what about for models like BERT that process the input as a sequence? These models don't view an input as simply a collection of isolated words; they are capable of picking out statements, sentences, or thoughts. Further, removing words from the input as a method to understand the model's predictions makes the

CONTENTS

input incoherent.


```
LIME defaults to masking
words, which produces a
type of incoherent
language that sequence
models like BERT were not
trained on.
```

What works well for a bag-of-words model seems unnatural for a sequence model. One could imagine that BERT works instead by treating input text as compositions of whole sentences, rather than compositions of words (although this is a simplification). In this case, it makes more sense to apply LIME at the sentence level. Removing sentences may still produce incoherent text, but if thoughts are relatively confined to sentences then this may be an acceptable trade-off.

☰ CONTENTS

For these reasons, we applied LIME at the sentence level for the final BERT model. For the NB-SVM baseline we explored both word- and sentence-level LIME.

Left: Word-level interpretability can be overwhelming and difficult to parse. Right: Sentence-level interpretability provides a more concise picture.

Regardless of the model, we found sentence-level LIME easier to digest. When highlighting individual words there is simply too much information to process.

Interpretability Provides Trust

With a model that can gauge the sentiment of a movie review, we can do some useful things. For example, we can visualize the overall sentiment of a particular film. This allows users to quickly gauge

☰ **CONTENTS**

ders to quickly gauge popular movies and could be used as a tool for recommending the best movies.

The Textflix dashboard makes it easy to find popular and unpopular movies.

Users can drill down to a particular title and read a sampling of its positive and negative reviews. But model predictions are rarely blindly accepted. Users who want to verify that the model is doing something sensible may opt to skim the reviews, looking for evidence of the model's predictions. This introduces a significant cognitive burden – reviews are long, but the sentiment may be expressed in just one sentence. The following review, where most of the text is simply summarizing

CONTENTS

text is simply summarizing the plot, is a good example.

Top: The model classifies this review as positive, but with no supporting evidence. Bottom: The model explains its positive classification by highlighting a single sentence, which should be enough to convince users to trust the prediction. View the review at on Texflix

Without interpretability, a user may need to skim each and every review for evidence supporting the model's predictions. But using LIME, we can automatically point to specific evidence that the model relies on to make its predictions. In this case, users can immediately see why the model has made a prediction and trust that it is correct.

☰ **CONTENTS**

Interpretability as a Summarization Tool

Interpretability tools that can point to specific evidence in the input to explain model predictions actually offer a new product opportunity. Specifically, we found that using LIME to identify highly polarized sentences

allowed us to construct high-level summaries of each movie's reviews.

20 positive review highlights
Nothing really unpredictable in this movie, but a solid flick in all respects. by elven_obscenity 2 days ago · Overall a really good movie with great performances from all the cast as well as the two leads, Mark Walhberg and Jennifer Aniston. by vaunted_tendency 6 days ago · Like a diamond, this movie shines. by elaborated_trout 11 days ago · I watched this movie the other night, and I have to admit, it was quite possibly the best film of this generation. by pliable_trauma 24 days ago · show all

6 negative review highlights
From the title change of Metal God to the 'safe' middle of the road 'Rock Star' to the lame soundtrack this movie plays out badly. by disdainful_folklore 1 days ago · Bad music (and I am a reformed eighties metal guy, so I would be vulnerable to some good stuff.) by freelance_proximity 30 days ago · There's way too much (boring) music in this standard formula-packed excuse for a movie and should be avoided for it at all cost. by essential_prospectus 88 days ago · But if you want to listen to good music I suggest spend the time looking at some concert recording with Bon Jovi, or Mötley Crue, it'll be more quality time. by registering_clothing 133 days ago · show all

```
By scoring each sentence
as positive or negative,
interpretability tools
enable summarizations of
sentiment for each movie.
```

These summaries give a reader a broad picture of others' sentiment toward

≡ **CONTENTS**

the movie, without them having to scroll through each of the individual reviews. Without interpretability, we could only show users entire reviews, which are burdensome to read. We find interpretability is an essential component of building interesting data products.

Successes and Failures

Transfer learning using BERT yielded impressive

results, especially considering the limited training data. It was a clear and significant improvement over the alternatives. But like any model, BERT has its limitations. In this section we examine some of the concrete successes and failures of the prototype model.

BERT Identifies Mixed Feelings

☰ CONTENTS

classification: positive · 94.1% certainty

by defaced_trauma · 61 days ago

Not the film to see if you want to be intellectually stimulated. If you want to have a lot of fun a the theater, however, this is the one. Lots of snappy banter(and some really cheesy banter, too). Mos Def and Seth Green are very funny as the comic relief. Exciting and creative heists and chase scenes. Mark Wahlberg and Charlize Theron(sexy)are appealing leads. And Donald Sutherland!

Interpretability at the
sentence level shows how
the model balances
opposing sentiments within
reviews. View the review
on Textflix

There are many reviews in the dataset that would classify as "easy" in the sense that they communicate clear sentiment using words that are positive or negative. The final deployed BERT model, unsurprisingly, handles these cases well. The passage above demonstrates some of this plain language, but is interesting in that it has individual sentences that communicate opposing sentiments. With the added interpretability mechanism, it is possible to not only show the model's overall prediction for the sentiment of the review but also its predictions for some of the individual parts. This

CONTENTS

helps draw attention to reviews that are mixed, where a single sentiment score does not necessarily tell the entire story.

BERT Handles Subtle Language and Negation Well

> "But Valley Girl is one of the few romance films I could sit through."
> – A review for *Valley Girl*

This review text is relatively subtle. It does not contain any explicitly positive or negative words, yet it is fairly obvious to a human that it conveys a positive sentiment.
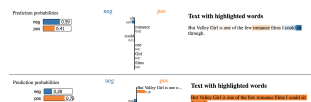
Top: BERT correctly predicts positive sentiment despite the indirect language. Bottom: NB-SVM incorrectly

☰ CONTENTS

predicts negative sentiment. Neutral words like "romance" and "sit" are associated with positive or negative feelings.

The BERT model produces the correct prediction, even if it is not clear how it infers this. The baseline NB-SVM model, on the other hand, must invoke simple rules which fail in this case. Using word-level LIME, we can see that the baseline model associates neutral words like "romance" and "sit" with positive or negative sentiment, even though these words on their own do not convey sentiment.

As a test of BERT's understanding, we can experiment with negating the entire phrase and observing the new prediction.

Top: BERT correctly flips its prediction when the "not" modifier is used to negate the positive

CONTENTS

statement. Bottom: NB-SVM does not recognize "not" as a negative word, and does not change its prediction.

Indeed, BERT now predicts a negative sentiment. The baseline model, however, can only rely on each word as a small piece of evidence. Even though it sees "not" as carrying negative sentiment value, the prediction overall is still positive.

There are cases where BERT fails, though, even with simple negation. Although LIME is a useful tool, it's not a catch-all. There is still quite some mystery in terms of how BERT works and when it will fail.

Sentiment Can Be Unclear

> "In any event, a very, very fun, but fairly bad, movie."
> – A positive review for *See No Evil*

☰ CONTENTS

"Average adventure movie that took a serious story and 'Holywoodised' it... The screenplay was average. The charm of Connery made up for his wrong Arabic accent and all the scenes with President T. Roosevelt were masterpiece takes."
– A positive review for *The Wind and the Lion*

"The storyline is okay at best, and the acting is surprisingly alright, but after awhile it's gets to be a little much. But, still it's fun, quirky, strange, and original."
– A positive review for *Don't Look in the Basement*

Each of these examples is from an instance where the BERT model made an incorrect prediction. Each of them is surprising, even to humans, in some way. That is, they each express

☰ CONTENTS

a statement of sentiment that is opposite of the rating the reviewer gave.

This underscores the fact that sentiment analysis is not really a binary task – in reality, humans may like some things and dislike others, which makes it somewhat nonsensical to distill their feelings into a single positive or negative classification. In some cases where the model gets it wrong, a human might also.

One encouraging aspect of these cases is that the BERT model is often quite uncertain in its predictions. Models will inevitably make errors, but the ability to provide a reliable measure of uncertainty makes it easier to use and trust their outputs.

Entity Assignment Presents a Challenge

classification: positive · 95.3% certainty

by razed_persona · 281 days ago

There comes a time in every big name actor's career when they get sloppy and accept projects that they wouldn't have touched with a 1000 ft. pole in their golden days. Remember "Taxi Driver"? *That was a fine film.* I can hardly believe that the De Niro of "Showtime" is the same actor. I would rather watch "Time Chasers" twice than see this film again. If anyone offers to take you to see "Showtime" or gives you free passes, or whatever, run away as fast and far as you can.

```
A review for Showtime
where the model is
confused by a positive
reference to another film.
```

This is a review for the film *Showtime*, a comedic cop film featuring Robert De Niro and Eddie Murphy. The review is quite negative, but features a mention of a separate movie - *Taxi Driver* - which the reviewer remarks is a "fine film." The BERT model does not recognize that this positive sentiment is actually attached to a different entity and should therefore be disregarded.

While BERT has been shown to be effective at the task of entity recognition, the fine-tuned model here was never explicitly taught to do this. In fact, since the model only has access to the review itself, it really has no way to know that *Taxi Driver* is not the subject of this review.

However, even if the model were given the movie title associated with each review, it is unlikely that it would learn such nuances. This is because it would not have nearly enough examples in its small training set for it to be able to learn this special case. Even including more examples might not be sufficient – instead, the training objective might need to be changed to explicitly ask the model to pick out entities and assign each of them a sentiment score.

This is an interesting example because it shows that even though transfer learning models may have the skills to perform particular tasks, they only invoke those skills if it is important for prediction accuracy during training. Transfer learning models are powerful, but they still fall far short of human intuition.

Product

Design

The final Textflix design.

We make prototypes to spark our clients' imaginations about how emerging technologies could be applied to their own business problems. Besides being functional, this means our prototypes need to tell a good story. In this section we'll discuss the design and storytelling decisions that went into Textflix.

Visualizing the Classification

Once we decided to focus on sentiment analysis and use the IMDB review dataset, the fundamental unit of the prototype was

CONTENTS

unit of the prototype was clear: the text of a movie review and the model's classification of that review as positive or negative. Presenting just those elements makes for a functional tech demo, but also a dry one.

The first step we took towards making things

more interesting *and* understandable was to apply the LIME interpretability technique at the sentence level. Using LIME, we show the user which sentences were driving the classification. This makes the classification feel much more dynamic. You can visualize (a simplified version of) the model's examination process.

Developing the Story

The second thing we did was develop a story about why these reviews needed to be analyzed. On the dataset side, we grouped

☰ CONTENTS

reviews under the movie or show they were reviewing, and selected the groups with the most reviews. Grouping reviews made it possible to build the prototype as an

imaginary movie/show review site.

The IMDB dataset, by design, covers a wide range of movies and shows, so sorting by the most reviewed gave us a rather eccentric list of entertainments. Part of the storytelling challenge of these prototypes is explaining dataset limitations like this one in a non-disruptive way.
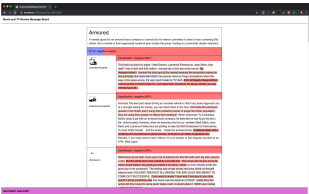
An earlier version of the prototype presented the reviews as part of an entertainment message board.

Originally, we thought

≡ **CONTENTS**

about imagining the reviews as part of a message board, which would help explain both the range of topics and the text-only nature of the reviews (movie review sites often have a mechanism for directly inputting user ratings, while message board

posts are typically limited to a text box). Later we realized we didn't have to be so specific to provide a plausible explanation for why the selection was so varied. Anyone who subscribes to a streaming service has seen the odd range of movies and TV shows that can result from behind-the-scenes license negotiations. We decided our make-believe service would be called Textflix, and that its weird selection was just a result of the licenses available.

Adding Drama to Analysis

Once we had our idea for a
fictional streaming site in

☰ CONTENTS

fictional streaming site in place, we began working on showing the value sentiment analysis could bring to the site. Here again we had to balance the design of a real product with the need for storytelling and a little bit of drama. We decided that you would be able to turn the sentiment analysis on and off in the prototype (in a real product there'd be no reason for the off option). This would help us emphasize, by contrast, the capabilities text analysis gives you. Without analysis, you must read each review, one-by-one, to get an idea of the overall sentiment of the reviews. With analysis you can see the general opinion at a glance.

Analysis makes sentiment computable and sortable, allowing you to answer questions like "what is the most liked movie?" (an indie drama called *What Alice Found* in this case).

The minimal design of the

CONTENTS

prototype emphasizes the new powers analysis gives you. With analysis off you're faced with long blocks of text and no color. With it on, you get green and orange sentiment indicators, underlines, and review highlights.

A Peek Behind the Curtain
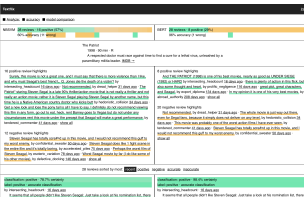
A movie page with the accuracy and model comparison options turned on. These options would not be appropriate for a consumer-facing product, but we included them for their explanatory power.

With analysis on, Textflix shows the capabilities that text analysis can bring to a product. We included two further analysis options to give people a closer look at the algorithm. These would

CONTENTS

not be included in a consumer-facing product, but we put them in the prototype for their explanatory power. The first option is "accuracy." It shows how well the model's classifications matched the review's original labels. With accuracy on you can see the model's overall performance as well as find reviews where the model got things wrong. This feature is obviously useful but it is only available because we are working with an example dataset that is fully labeled, something unlikely to happen in real life (if your real life dataset is fully labeled you don't need a model to

classify it). The accuracy feature in the prototype, then, is a peek behind the curtain. It wouldn't be possible in a real product, but in Textflix, seeing the errors helps build your intuition about the model's performance and where it might tend to get

CONTENTS

where it might tend to get things wrong or right.

The second feature is "model comparison." When activated, this feature opens a split-screen view, where our transfer-learning trained model is on the right, and NB-SVM, our baseline model, is on the left. Seeing their classifications side-by-side gives you a concrete idea of the gains made through transfer learning. You can compare the accuracy of both approaches and see reviews where they disagree. In a final consumer-facing product, you'd want to only expose the best model (though checking it against a baseline during development is still very much recommended).

There were more options we could have added, but we wanted this prototype to be clearly focused on a core set of ideas. Most importantly, we wanted to show the kinds of

capabilities that text analysis could open up for a product. Then we wanted to provide a view into the model's accuracy and show it in the context of a less advanced approach. We hope the final product will be both inspirational and informative for people working on similar projects.

≡ **CONTENTS**