Applying Natural Language Processing to the World of Wine

Tim Aiken, Clara Meister
{timaiken, cmeister} @stanford.edu

1. Abstract

Wine reviews present an interesting problem within the natural language processing space. Most wines have a textual review of them, which can provide insights into the profile of a wine, but which can also heavily abuse industry specific terminology. We used a dataset of 100k wine reviews to examine the efficacy of NLP on wine-specific language.

We trained a GloVe encoding on our specific dataset to encode semantics of wine speak, and then trained two models, a bidirectional LSTM to predict varietal from text input, and a LSTM to predict text output from varietal and point score input. Overall the models performed much better than expected, with high accuracy on varietal prediction and qualitatively convincing results on text generation.

2. Introduction

Wine is a notoriously convoluted and veiled industry, making it extremely hard for people with little knowledge to enter it confidently. While a consumer may have an idea of characteristics they'd like to see in their wine, they may not have any knowledge of which grape varietals from which regions would exemplify these characteristics. Someone may then turn to a wine description or review in the hope of gaining some insight about the bottle their holding in their hands. Unfortunately, these reviews are often nonsensical to the average person, using flowery language with strange descriptors. We attempted to use two deep learning models to extract concrete information from these imprecise-sounding and hard to read reviews.

Our first model was a bidirectional LSTM to predict wine varietal (i.e. Chardonnay, Bordeaux Blend) from a wine review. The model took in 200-dimensional encodings of the words in the review and outputted probability predictions for the 31 classes of varietals.

The second model was a LSTM to predict review text from varietal and point score. The model took in a concatenation of the varietal/point score and the current word of the review. Then, at each time step, it would predict the next word in the review.

3. Related work

Little work has been done on wine as a specific subset of NLP, but a few papers can be found. I. Hendrickx et. al. [1] scraped a dataset of 70k reviews from Wine Spectator for the purpose of guessing wine characteristics from wine reviews. They trained Word2Vec encodings and then used an SVM to predict whether the wine was red, white, or rose, producing f1 scores between 78 and 98.

In a similar project a pair of Stanford students in CS224U [2] scraped 130k wine reviews from twitter and then attempted to predict characteristics from the reviews and reviews from characteristics, both using LSTMs. Their varietal prediction LSTM produced a test accuracy of 53% while a Naive-Bayes baseline achieved an accuracy of 45%. The text-generation LSTM did not produce qualitatively compelling results.

In a broader sense LSTM's were shown in their original paper to be good for long text classification [3] and have been repeatedly proven to be useful for text generation, especially for tasks such as image captioning [4], [5].

The most well-known word embedding method may be Word2Vec by Mikolov et. al [6] which uses word sampling to produce similarity information. More recently a Stanford research project named GloVe [7] showed that co-occurance matrices can be used to train high performance embedding vectors.

4. Dataset and Features

The dataset we've chosen to work with was scraped by Zack Thoutt from the Wine Enthusiast website [8] and contains 120k data points (after duplicates have been dropped) split into 100k training/10k validation/10k test. We chose the top 30 most represented categories in our dataset along with an 'other' category that the rest of the examples fell into. These categories included "Chardonnay", "Cabernet Sauvignon", and "Bordeaux-Style Red Blend" among others. The majority of data points can be classified as one of these 30 varietals, with the remaining 19% classified as "other." From each datapoint we used the following categories:

- Variety (the type of grapes used to make the wine)
- Description (multiple-sentence written review)
- Points score (80-100)

To preprocess the data, we stripped the description variable of all special characters (i.e. transforming ű -> u), transformed all words to lowercase, and added start and end tokens to the sentences.

Sample data point:
Country: france
Description: big structured tannins lie over the ripe fruit the wine has a dense character the young tannins dominant given a year the fine fruit will come through to give a sweeter blackberry character
Province: bordeaux
Variety: bordeaux-style red blend

To feed our descriptions into an RNN we had to come up with a word encoding method that would be reasonably low dimension and would contain information about the way words are used in wine reviews. To accomplish this we trained our own GloVe matrix off of our training data.

We constructed a co-occurrence matrix of words in our corpus and then trained 200-dimensional word encodings using the GloVe algorithm. However, because our corpus is relatively small, we hot-started our encoding by initializing the weights to be a GloVe matrix trained on all of Wikipedia (available through the GloVe research distribution [7]). This allowed our encodings to start containing information about English in general and then specialize to the language used in wine reviews.

A t-SNE projection of the 200d GloVe encodings can be seen in Fig. 1. Words close to each other in the visualization are close to each other in the 200d GloVe space, and thus are considered "similar" in our encoding. Some of the denser clusters are types of wine grapes (pinot, syrah, tempranillo), countries (france, portugal, turkey, russia), and descriptors (complicated, delicious, delightful).

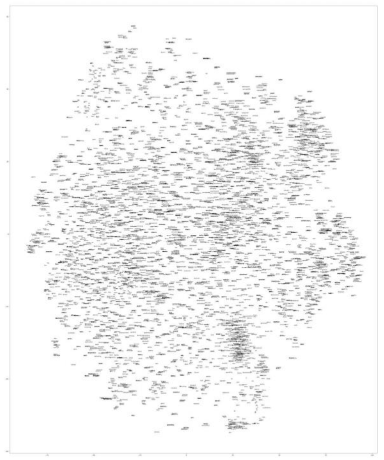

Fig 1. t-SNE visualization of GloVe encodings

5. Methods

In short, our problem was to take a wine review (string of text) and output a class (one of 31 categories). Because the wine reviews had a sequential structure, we chose to narrow down our model search to Recurrent Neural Networks. Within this domain, we explored LSTM (long short-term memory) and GRUs. Because our classification relies heavily on how the words in a sequence are laid out, we needed a model that would transmit data from previous states in order to properly calculate the impact of an input word. The ability of an LSTM to avoid long term dependencies while learning the 'impact' of previous states in the sequence made it an ideal model for our classification problem. Bidirectionality was a useful feature since words later in the sequence also had an impact on the weight of an input word. In our problem, we had the entire string of text immediately so were able to look at the sequence both backwards and forwards. Ultimately, a Bidirectional LSTM was chosen for the classification problem because of its performance over the simpler GRU model.

The model used for the varietal classification problem was a Bidirectional LSTM with 100 hidden nodes, which was then fed into a Dense layer with 100 hidden nodes and 30% dropout rate, which was finally given to a Dense layer of hidden size 31 with softmax activation. We used the output of our last layer to classify an example into one of 31 categories (30 varietals and 'other'). This LSTM was trained using categorical cross-entropy loss and an Adam optimizer. Performance was measured in accuracy (percentage of varietals it categorized correctly).

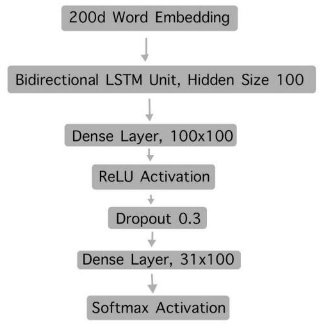

Fig 2. Varietal Classification LSTM parameters

The wine review generation model is a single-layers LSTM with hidden size of 512. An LSTM was chosen over an RNN or GRU because of its potential for memory retention. To output reasonable sounding reviews we needed a model that could avoid repetition and build on phrases generated many words ago.

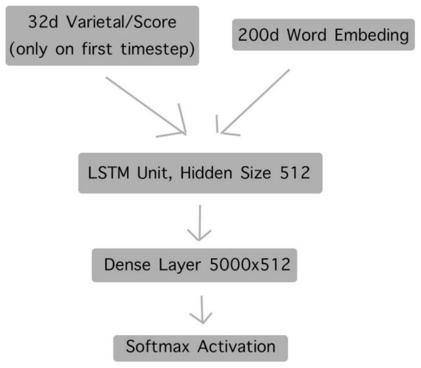

Fig 3. Text Generation LSTM parameters

The LSTM took as input a concatenation of two vectors: the first contained a one-hot encoding of the varietal (first 31 values) and the point score (the 32nd value) while the second contained the 200d encoding of the current word. The varietal/score vector only contained values on the first time step and was zero at every other time step, which is a similar strategy to that employed by image captioning systems.

After progressing through the LSTM layer the activation is then passed through a dense layer and a softmax activation function to produce a 5000 dimensional output (the size of our vocabulary) that contains the probability that each word in the vocabulary is the next word in the review.

This LSTM was trained using categorical cross-entropy loss and an Adam optimizer. Performance was measured in accuracy (percentage of time the next word is guessed correctly) and perplexity, which measures how well a probability distribution predicts a sample. The formula for perplexity is:

$$2^{\frac{-1}{m}\sum log(p(s))}$$

Which intuitively is two raised to the mean of the categorical cross entropy.

To generate text from the LSTM we used beam search. Beam search is essentially an adaptation of breadth-first search that runs in constant time but may not find the optimal solution. At each time step you have *b* words to test. From those *b* words you choose the *b* most likely words that could follow them. These *b* words would then be fed through the LSTM to generate the options for the next time step.

6. Experiments/Results/Discussion

The varietal prediction model was trained for 20 epochs with a batch size of 256. We performed a grid search over several different parameters in order to choose the best model. The performance of smaller LSTMs with 20 or 50 hidden states plateaued at about 60% validation set accuracy while larger LSTMs with 200+ hidden states had comparable or inferior performance to a model with 100 layers. Other optimized parameters included batch size (64, 128, 256), learning rate (0.01, 0.005, 0.001, 0.0005) and dropout rate (0.2 - 0.5). We also tried adding an additional layer between the LSTM and output layer, which gave us an increase in accuracy of ~1% on the validation set.

Other models we explored included GRUs (the best of which provided an accuracy of ~55%), stateful LSTM models, and Convolutional LSTM models. The most promising of these other models were the Convolutional LSTMs, which added a 2D convolutional layer followed by a 2D max pooling layer. This model was based off of the work of Zhou, et al. in their research on “Text Classification Improved by Integrating Bidirectional LSTM with Two-dimensional Max Pooling.” Ultimately, the accuracy of these models on our validation set was either comparable to or worse than the simpler Bidirectional LSTM model. Following the principle of Occam’s razor, we went with the simpler model.

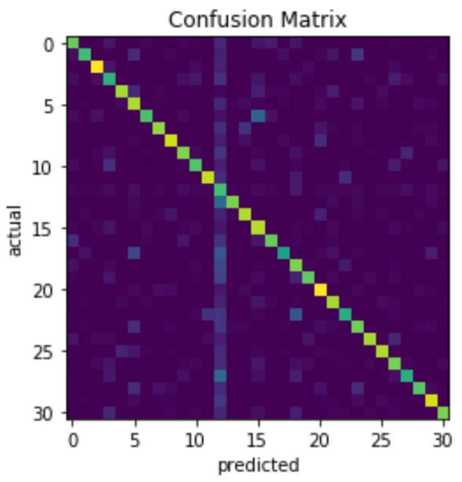

Fig 4. Varietal Classification Confusion Matrix (by percentage)

Our test set accuracy was 63.43%, which beat the next best known model by ~10%. Recall for certain varietals was much higher than for others. For example, the model correctly classified 82% of Chardonnay while only correctly classified 20% of Tempranillo (43% of Tempranillo was classified as other, hinting that Tempranillo might have very varied characteristics). Other sizeable mistakes were that 25% of Gamays were classified as pinot noirs, which is an expected result since the grapes have very similar profiles. Finally, despite the ‘other’ category representing ~19% (the biggest class) of the data, our model only classified 20% of the test set as ‘other’, indicating it did not default to the largest classification to increase accuracy.

Overall, many of the misclassifications made by the network were mistakes commonly made by sommeliers or other wine experts. In cases where the predicted varietal was distinctly different from the actual varietal, such as misclassification of Gamay as Bordeaux-Style Red Blend, the description often subjectively seemed more fitting for the predicted varietal (i.e. “this offers plenty of tannins and a dry firm structure along with the weight of ripe plums blackberries and a rich character the wine is certainly going to age with its solid texture and bold fruit aftertaste drink from” contains more standard characteristics of Bordeaux than Gamay even though this comes from a Gamay example).

The text generation LSTM was trained for 10 epochs (each epoch took about 20 minutes) with a batch size of 200. This batch size was chosen to provide a good compromise between the quick learning of a stochastic training method and the efficiency of normal gradient descent. We did not tune the Adam optimizer as its adaptive learning rate strategy means any tuning of the start learning rate gets nullified fairly quickly. We manually tested hidden layer sizes between 50 and 1000 and saw significant performance improvements as the number of neurons increased. A size of 512 was chosen as a compromise between accuracy and time to train.

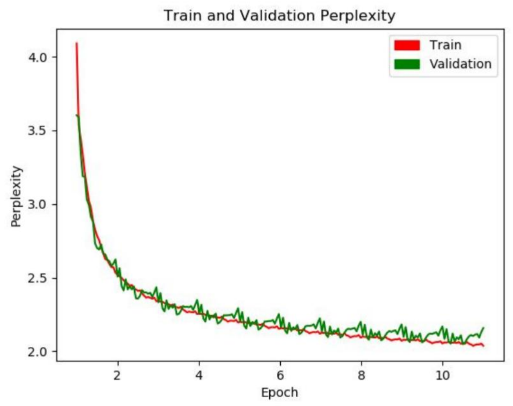

Fig 5. Perplexity for Text Generation LSTM

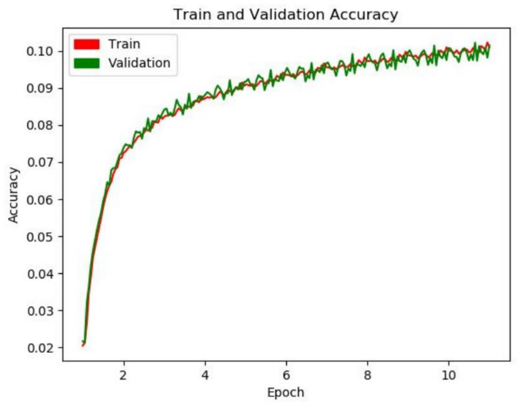

Fig 6. Accuracy for Text Generation LSTM

As you can see in Fig 5 the perplexity of both the training set and validation set decreased over time. Similarly, the accuracy of the training and validation sets (Fig 6) increased over time. Both graphs show that the LSTM is not overfitting as the validation set does not do worse than the training set. Final results for the model can be seen in Fig. 7.

	Train	Validation	Test
Accuracy	0.101	0.102	0.095
Perplexity	2.037	2.158	2.157

Fig 7. Results for Text Generation LSTM

However, for a text generation LSTM, accuracy doesn't say too much about the true performance, except to confirm the model isn't overfitting. Qualitative analysis of generated outputs is a better metric for how good the model may be. Some examples of outputs from the LSTM can be seen below. As a reminder, punctuation was stripped from our training data so the output will also be devoid of it.

Input: Red Bordeaux Blend, 100/100 pts:
Output: with firm fruit aromas that give concentration a very fine margaux the second label this estate that are currently in to bordeaux blends cabernet cabernet and the grape of merlot cabernet petit malbec malbec is complex in a wine its complex but the core are there black fruits licorice chocolate black plums dark and spicy smoky cedar (...etc)

Input: Pinot Noir, 80/100pts (bad score):
Output: very sweet almost syrupy a pinot thats the acidity accentuates and its soft which doesnt get there its that will the raspberry cream note that taste overwhelm it this for those tannins for aging yet this rich too much like what it has more power in their bottles sh and are best at and try holding down

Qualitatively these results are very encouraging. The Red Bordeaux Blend review outputs sentiment ("very fine", "complex") and understands flavors common in that type of wine ("spicy smoky cedar", "black fruits"). Similarly, the Pinot Noir review has sentiment ("almost syrupy", "doesnt get there") and flavors as well. It appears that the LSTM was able to learn the distinctions between different types of wine and different qualities of wine, which is quite encouraging.

7. Conclusion/Future Work

Wine reviews are notoriously imprecise and flowery, but in the end our models were able to extract meaningful information from the chaos. Our varietal prediction bidirectional LSTM achieved an accuracy of 0.65 on 31-class predictions while our text generation LSTM produced qualitatively compelling outputs and reached a perplexity score of around 2.0. The simplest Bidirectional LSTMs worked best in both situations, indicating other models had over or under-fitted the data.

With more time we would expand our varietal prediction model to predict other characteristics of the wine: point score prediction, region prediction, and author prediction could all be interesting extensions. For varietal prediction, we hope to try changing the loss metric (for example, penalizing incorrect categorization of a red wine as a white varietal higher than of a red wine as the incorrect red varietal) or changing the classification schema. Similarly, we would increase the dimensionality of the input to the text generation LSTM to see if more specific and accurate reviews could be generated without overfitting.

8. Contributions

Clara:

- Coded, tuned, and trained the Bidirectional LSTM for varietal prediction
- Parsed, cleaned, and split the dataset into train, test, and validation chunks

Tim:

- Coded, tuned, and trained the LSTM for text generation
- Trained the GloVe encodings from a co-occurrence matrix based off our text

9. Code

Code for this project can be found at: https://github.com/cmeister747/cs230

10. References

[1] Hendrickx, I., Lefever, E., Croijmans, I., Majid, A., & van den Bosch, A. (2016). Very quaffable and great fun: Applying NLP to wine reviews. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (Vol. 2, pp. 306-312).

[2] Robson, F., & Amdahl-Culleton, L. Classy Classification: Classifying and Generating Expert Wine Review.

[3] Sundermeyer, M., Schlüter, R., & Ney, H. (2012). LSTM neural networks for language modeling. In *Thirteenth Annual Conference of the International Speech Communication Association*.

[4] Potash, P., Romanov, A., & Rumshisky, A. (2015). Ghostwriter: using an LSTM for automatic RAP lyric generation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (pp. 1919-1924).

[5] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., ... & Bengio, Y. (2015, June). Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning* (pp. 2048-2057).

[6] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111-3119).

[7] Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532-1543).

[8] Z. Thoutt: https://github.com/zackthoutt/wine-deep-learning