Classification of Drop-Shipped vs. Authentic Product Listings via NLP-Based Classification Techniques

Winston Yin
NYU Stern
wy2145@nyu.edu

Pranav Kanchi
NYU Stern
pk2296@nyu.edu

Kim Young
NYU CAS
kay271@nyu.edu

Jonathan Simonsen
NYU CAS
js14111@nyu.edu

Abstract

The shift towards E-commerce has driven growth in a deceptive practice known as "drop-shipping," where individuals market generic and wholesale products as their own, misleading consumers into paying inflated prices for subpar goods while also introducing a number of product safety issues. In response, this paper employs various natural language processing (NLP) methods to develop a tool capable of efficiently and accurately distinguishing between the textual content of drop-shipped and authentic product listings. This paper outlines a comprehensive methodology, including data collection and processing, to train and evaluate NLP classification techniques. To construct a working dataset, we created a web scraper to extract product listings from online websites for both drop-shipped and authentic products. We employ several feature extraction and vectorization methods such as Word2Vec and TF-IDF to convert textual data into regressable vectors, upon which we test the effectiveness of logistic regression classifiers. We also utilized more advanced encoder models such as ALBERT and RoBERTa to classify drop-shipped products, demonstrating a diverse approach to address the complex challenges posed by drop-shipping. Our results show that more complex context-driven models derived from BERT are highly effective at distinguishing between drop-shipped and authentic product listings and have the potential to serve as the basis of a dropshipping product classifier.

For a further information on our data procurement and classification methods, we have attached a repository containing all of our code here.

1 Introduction

With the rise of the internet as well as the build-out of a nationwide fulfillment network, traditional brick-and-mortar retail has given way to the rise of E-commerce, enabling brands and products to be sold in a completely digital manner. While this has greatly increased the convenience of shopping for the consumer, it has also allowed for savvy individuals to purchase and market generic products from a third-party supplier as their own, misleading consumers into overpaying for lower-quality products, a practice known as "dropshipping". This especially grew post-COVID, with social media formats like TikTok and Instagram Reels allowing for these dropshipped "brands" to quickly advertise their product to a mass audience. We aim to utilize a variety of natural language processing methods to create a tool to distinguish between the textual content of "drop-shipped" and authentic product listings in an efficient and accurate manner.

Classification starts with vectorizing product listing data into word and text embeddings that can then be used in classification algorithms. We use existing embedding libraries and feature extraction methods such as Word2Vec and TF-IDF to construct a set of embeddings from our product listings. These embeddings are then used as inputs for a multitude of different classification approaches, including logistic regression. We also applied state-of-art methods in the form of BERT (Bidirectional Encoder Representations from Transformers) and other derived models to explore the impact of more context-based models on classification performance. For the more advanced BERT-based models, we did not need to create embeddings or features ourselves, as they can classify and encode text. For each approach that we tried, we fine-tuned the data, embeddings, and algorithms used through an iterative optimization process, and then finally compared their performance to each other through common metrics such as precision, recall, and F1-score.

2 Problem Statement

The dropshipping business model harms both customers as well as the online sellers of credible, branded products. Firstly, it deceives customers into overpaying for products of perceived "premium" quality, which may also fall outside of standard product safety standards. This not only leads customers to waste money on below-par goods, but can cause injuries through product malfunctions (i.e. faulty appliances). In addition, dropshipping also fosters distrust in e-commerce platforms and authentic online sellers, discouraging online purchases and reducing a business' top and bottom-line. As such, there is a vested interest for both e-commerce platforms and consumers to possess a tool that can accurately distinguish between the product listings of drop-shipped and authentic products based on the nuanced differences between listing information such as return policies, pricing information, and product descriptions. This tool would allow for customers to pay the right price for the proper product while also promoting the continued development of e-commerce, which are both goals that we would love to be able to support.

3 Existing & Related Literature

Due to the sudden rise of dropshipping over the last five years, very little existing NLP research dives into the express topic of classifying different product listings. However, a significant amount of existing work has been done on the potential merits of various NLP techniques for classifying other text bodies, particularly in tasks such as analyzing social media posts for fake news or propaganda.

The starting point of any NLP-based classification technique is feature creation and selection, the conversion of textual bodies into a more limited and representative set of features, upon which a classification algorithm can then be applied. A well-studied sentence vectorization/feature extraction method that we also learned in class is TF-IDF (Term Frequency - Inverse Document Frequency), which distills a word or term into a numerical measure of its relative importance in a body of documents. Despite being a relatively simple algorithm, TF-IDF has been shown to guard against the skewing of results from a high level of class imbalance. By testing various algorithms including logistic regression and support vector machines (SVMs), Christian Padurariu et al. (2019) (1) found that simple text representations such as TF-IDF, work better for text datasets with large imbalances rather than complex embedding methods (i.e. GloVe embeddings) do. This is incredibly important when classifying dropshipping data, since the magnitude of available original listings is likely much greater than that of drop-shipping listings. Another more advanced and modern method of feature selection is word2vec, a method, introduced by Mikolov et al. (2013) (2), of representing words as a set of continuous vector spaces, that has been found to capture the semantic relationships and similarities between words, something that TF-IDF lacks. We intend to utilize these two feature selection methods as the basis of transforming product listing data into a set of classifiable features.

The other component of classification is the actual classification algorithm itself. In this case, another very foundational technique, logistic regression, serves as a solid starting point. In a study by Li et. al (2019) (3), researchers looking to classify propagandistic sentences by leveraging TF-IDF to vectorize social media posts and extract the most relevant features. From there, they were able to successfully apply a logistic regression classifier against the top 100 features ranked by TF-IDF along with several selected emphatic content features. However, it is important to note that Padurariu et. al (2019) (1) also found that linear classifiers (such as logistic regression) are more biased in the context of significant class imbalance, which is something that should be remembered in the course of working on this project. A more advanced angle that has been taken here is the application of encoder-based models such as BERT. The BERT paper authored by Chang et al. (2019) (4) was a seminal paper that utilized the advancements in transformer architecture mentioned in Vaswani et al's (2017) (5) seminal "Attention Is All You Need" paper. The use of transformers to allow for self-attention vastly allowed for leaps forward. The paper is largely a foundational one but points to the possible increases in accuracy should we use BERT or some of its more recent innovations (ALBERT or RoBERTa). BERT, as an encoder-based model, performs extremely well for classification-based tasks, specifically having better semantic representations of context in long sequences through the self-attention mechanism. It also automates the selection and creation of feature vectors, eliminating the use of either word2vec or TF-IDF.

In direct tests against other, more basic algorithms for use in the classification of informative Covid-19 tweets, Babu and Esawri (2020) (6) found that BERT-based models performed meaningfully better than simpler TF-IDF and support vector machine (SVM)-based models, achieving F-scores near 10% above its simpler counterparts. These encoder models could also be applied to our current use case and evaluated against simpler models such as logistic regression to determine the role of complexity in model accuracy, and understand if it is truly necessary to utilize the cutting-edge.

4 Data

4.1 Data Collection

To obtain a dataset of drop-shipped product listings in lieu of a pre-existing database of listings, we wrote a Python-based web scraper to collect product listing data from AliExpress, a popular site for drop-shippers to obtain original product copies that they would then rebrand. We identified three drop-shipped products of differing levels of popularity to better represent the greater drop-shipping space, LED lights (low popularity), lightsabers (high popularity), and power adapters (medium-popularity), and extracted roughly 800 relevant product listings from AliExpress for each category. By limiting our scrape to particular types of products with a certain product category, we would be able to standardize away the confounding variable of product types, as well as select products that were easier to find drop-shipped product listings for. While this does include the possibility of the dataset not generalizing well to other product types, we felt that the added confidence we would have in identifying drop-shipped products as well as the fact that drop-shipped listings are fundamentally very similar across categories made the trade-off worthwhile for us. From here, our web-scraping program pulled a variety of product listing information, including review content, pricing, and seller descriptions, before consolidating all of this information into a single entry per product listing.

We then repeated a similar process to build a set of drop-shipped products, applying our web scraping program on known dropshipping brand websites sourced from communal dropshipping brand lists on social media platforms such as Reddit to search for the same products and listing information. Given the lack of certified drop-shipping brands, we collected much less data here, roughly 150 entries per category, giving a final split of 450 (15%) drop-shipped product listings and 2,456 (85%) authentic product listings.

4.2 Data Processing

To obtain a dataset of drop-shipped product listings in lieu of a pre-existing database of listings, we wrote a Python-based web scraper to collect product listing data from AliExpress, a popular site for drop-shippers to obtain original product copies that they would then rebrand. We identified three drop-shipped products of differing levels of popularity to better represent the greater drop-shipping space, LED lights, power adapters, and lightsabers, and extracted 800 relevant product listings from AliExpress for each category. By limiting our scrape to particular types of products with a certain product category, we would be able to standardize away the confounding variable of product types, as well as select products that were easier to find drop-shipped product listings for. While this does include the possibility of the dataset not generalizing well to other product types, we felt that the added confidence we would have in identifying drop-shipped products as well as the fact that drop-shipped listings are fundamentally very similar across categories made the trade-off worthwhile for us. From here, our web-scraping program pulled a variety of product listing information, including review content, pricing, and seller descriptions, before consolidating all of this information into a single entry per product listing.

After we cleaned the data, we then placed it into a Pandas dataframe for training, development, and testing of various NLP classification techniques, and also shuffled the overall dataframe. From here, we implemented k-means cross-validation via the sci-kit learn library to split the data into 3 equal sections. Each section would have equal proportions of drop-shipped vs. authentic products in order to maintain a balanced representation of classes, which is especially crucial when dealing with imbalanced class distributions. With regards to labeling, we mapped the classification labels from our scraped data to numerical values, with drop shipped product websites represented as ones and authentic product listings represented as zeros.

4.3 Data Limitations

One specific limitation of our data was a lack of diversity in the platform in which we sourced drop-shipped products. We sourced them all from a number of different sellers on AliExpress, which does not consider sellers on other e-commerce platforms such as Amazon, or drop-shipping brands that host their own websites. This could limit our model's generalization capability to more diverse dropshipping practices. This problem could be solved by building separate web scrapers to parse through a number of different websites and their product listings, which we intend to do to further the application of our project, but is too time-intensive to do in our current time window. However, we believe that this will produce diminishing returns, as drop shipped products are relatively similar across platforms. Further, since we have removed website structure and many dropshipping brands copy terms and descriptions across platforms, it seems likely that our accuracy would persist: comparing our current data to a quick manual scrape of Amazon and eBay product listings shows very high (0.8+) cosine similarity scores between the two groups.

Another shortcoming of our data collection methods would be the risk of mislabeling. Depending solely on social media posts for identifying drop-shipped products introduces the risk of mislabeling. Some products identified as drop-shipped may not actually follow the dropshipping model, leading to inaccuracies in the dataset and potentially biased model outcomes. This is a difficult problem to surmount, as there is no universally accepted dataset for drop-shipping brands, and thus, our best approach was to rely upon social media to crowd-source a consensus opinion vetted by multiple parties.

5 Methodology

5.1 Text Embeddings and Vectorization

We used two different methods of creating vectorized feature sets from the data we collected: TF-IDF as well as the context-driven Word2Vec algorithm created by Google.

Our first approach to creating a feature set for our dataset was through TF-IDF. We treated the universe of words/features as the data entries that we scraped previously, and computed the TF-IDF scores for each of these words before vectorizing the output. From here, we modified the vectorized output to only utilize the top 1000 features by term frequency. This would allow for a model to be trained purely on the relative frequency of specific keywords in drop-shipped and authentic product listings.

Transitioning to Word2Vec, we adopted a different strategy, moving beyond the frequency-based representation. Word2Vec represents individual features as dense vectors in a continuous vector space, capturing semantic similarities and relationships. Mirroring the approach with TF-IDF, we trained a Word2Vec model on the training segment of each fold. In our specific vectorization of product listings, we chose to represent process word as a feature, with our final vector composed of 100 dimensions in total. The resulting word embeddings can then be used to train models that understand the semantic nuances and associations within the text data, offering a more sophisticated representation compared to traditional frequency-based methods like TF-IDF.

5.2 Logistic Regression on Vectorized Data

We then utilized logistic regression to help us classify the text. To apply logistic regression to the feature vectors we created using TF-IDF and Word2Vec, we utilized sci-kit learn to test a number of different variants of logistic regression, including a class-balanced regressor and a regularization-tuned model to limit overfitting, training and evaluating each model on every fold created during our data collection process. In preliminary tests, we noticed that applying a class-balanced version of logistic regression performed the best, likely because it remedied a large portion of the problems created by the large class imbalance between drop-shipped and authentic products in our database. As such, we focused our time on refining the class-balanced logistic regression model over the others, and found that the best model set-up would be to train and validate the model through 3-fold cross-validation, which served as the perfect balance between performance on existing versus new product listings.

5.3 BERT-based Models

We then wanted to approach the task with some more powerful models, and looked to the BERT framework, including base BERT, ALBERT, and RoBERTa, to do so. Since BERT-based models may be more sensitive that logistic regression models to class imbalances, we modified our existing split dataset by randomly selecting 60% samples form the non-dropshipped category and 40% from the dropshipped category. We chose this split as in practice it showed promise as a good tradeoff between having a large enough data sample, as we had less dropshipped product examples, and not over-representing one class.

We also utilized stratified K-fold cross-validation on the training/development dataset with

3 splits. We chose this value simply due to time training constraints for the model, but further versions of the paper could explore having a larger number of folds. Using this approach ensures that each fold of the dataset has the same proportion of each class, ensuring the balance achieved in the preprocessing step. Further, the K-fold stratified cross-validation proves to better utilize our limited dataset and reduce bias and variance. This was important for us due to the previously mentioned dataset issues, and showed significant model improvement when implemented.

We first used the base BERT model with a binary classification head, due to only having two classes. We implemented some configuration adjustments. First, we wanted to ensure fair penalization of misclassifications, and such we computed and applied class weights to address any possible residual effects of the 60/40 split. We also trained the model using the AdamW optimizer with a learning rate of 1e-5 and a weight decay of 0.05. These hyperparameters showed promise through manual testing. Finally, the model was trained over a total of 10 epochs for each fold, with evaluation metrics calculated at the end of training.

We then repeated the process for other BERT models and on different datasets. We ran BERT, RoBERTa, and ALBERT on the dataset of all three product categories. We also separated the data by product category and ran the ALBERT model on the individual categories as well.

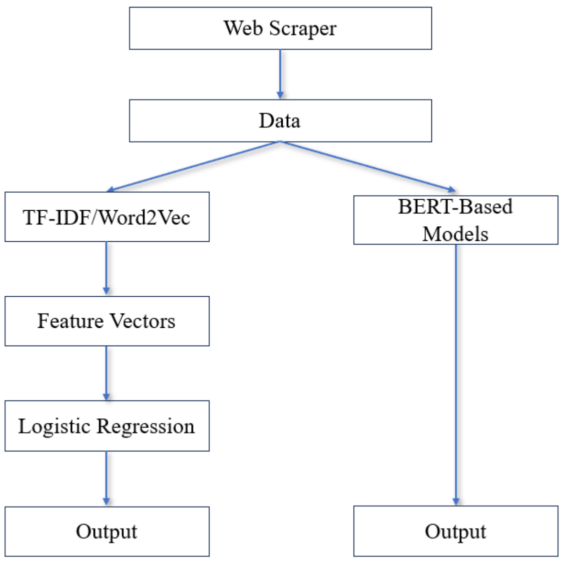

Figure 1: Model Process Flow

5.4 Model Evaluation Criteria

Post-training our models, we assessed model performance through metrics-like accuracy, precision, recall, and F1-score. These metrics helped us understand how the model performed for our problem. Our product is meant to alert the public, and there is very little downside to a false positive, just lost time, compared to a false negative, potentially meaning significant loss of money. Thus, we wanted to maximize recall, so as to err on the side of warning the consumer. We also looked at the confusion matrix for error analysis to help inform us on what optimizations we should make. Based on the performance metrics that we obtained from each training iteration, we then made changes accordingly to improve the performance of each of our models.

6 Results

6.1 Overall Performance Metrics

	Acc.	**Prec.**	**Recall**	**F1**
TF-LR	0.943	0.946	0.650	0.770
W2V-LR	0.967	0.945	0.712	0.809
BERT	0.999	1.000	0.990	0.994
RoBERTa	0.999	0.990	1.000	0.995
ALBERT	0.999	1.000	0.997	0.998

Table 1: Performance Metrics

Table 1 details the results we obtained from the various different classification setups that we tested. Overall, it can be seen that the more advanced and holistic a model we use with regards to feature creation and classification, the better the results. We believe that this is because the differences between drop-shipped and authentic product listings likely lies in the context of the text of each listing, such as the presence of various promotional phrases and search terms strung together to maximize the search ranking of a product.

6.2 Results Analysis

In comparing the difference in performance (from Table 1) between logistic regression models run on TF-IDF vectors versus one run on embeddings created by Word2Vec, the accuracy and precision remain very similar, but the recall for drop-shipped items increases significantly. This is likely because Word2Vec, by creating a multidimensional that considers context across a textual body, picks up on some added context-based elements not present in TF-IDF to better identify a number additional drop-shipped product listings at a comparable rate of accuracy. However, both systems are still relatively conservative as a whole when it comes to flagging drop-shipped product listings, which may be a function of the class imbalance present in our scraped dataset, which skewed massively in favor of authentic product listings.

Moving to the more complex models, we found a series of interesting conclusions from our confusion matrices and our error analysis. Firstly, the model worked best when utilized to predict the outcome of the same product it trained on. We noticed this through the testing of many new datasets that were sourced from similar websites (i.e. AliExpress for the non-dropshipped and assorted websites for the dropshipped) and the models failed somewhat to generalize, over multiple iterations. We speculate from this result that this means the model is understanding the product descriptions, and is capable of understanding from description whether a product is dropshipped or not. While this result was disappointing to know in terms of sheer results, it was nice to know this from a technical understanding. This meant that it was less dependent on the website structure itself, one of the key worries we had despite removing all remnants of the HTML code. This should mean that the models are resilient to new websites and any model trained on a product generalizes well to any new website structures, simply by understanding the text descriptions of the products. From a consumer perspective, we came to the conclusion that this was optimal, as for an actual use case, we could guarantee performance for any products we mention we cover. This also menas that we will need to train on a larger set of products before utilizing this model for general classification tasks across multiple new product sets.

We also found a balance between compute efficiency and performance results. One of the concepts we were worried about was time to train, simply because of the multiple techniques that increased the complexity of the already advanced BERT-based models. Similar to our understanding from the literature review, the advanced encoder models outperformed significantly, due to the encoder structure better understanding semantics than the simpler Word2Vec and TF-IDF models. This however, came with the tradeoff of more runtime, as, for example, the Word2Vec model took roughly 3 mins on CPU whereas the base BERT model took around 23 minutes on a T100 GPU. We discovered that ALBERT provided a good balance between the benefits of the more complex models due to using the encoder model structure and still having a reasonable compute time. Hyperparameter modification further allowed for reasonable compute and allowed us to have powerful models that could be trained relatively quickly on a GPU. This was a key point of analysis for us due the sheer size of the dropshipping problem. Understanding either the compute cost of either running a model per product or one large model on an extremely large dataset, we wanted this to be reasonable so that we could push this out to a consumer audience and help protect against dropshipping at scale.

7 Conclusion

In summary, our study aimed to explore multiple NLP-based classification setups to find one that would work best as a classification tool for identifying drop-shipped product listings. We explored the results of applying logistic regression to vectorized features extracted using TF-IDF and Word2Vec as well as modern encoder-based models derived from BERT. Through our tests, we discovered that BERT and its related models performed meaningfully better than logistic regression-based approaches, and reached levels of performance where real-world application is a possibility.

Though the generality of our code may be limited by our data collection methodology (primarily class imbalance and sourcing), we believe our findings demonstrate that with further refinement with regards to data procurement and algorithms, creating a NLP-based tool for identifying drop-shipped products online is a very feasible task. We hope that our work can be built off of by future researchers, and that a working tool can be provided to customers and e-commerce platforms in the near future, promoting fair play and improved safety standards in the world of online shopping.

8 Future Work

Going forwards, we see several areas of improvement for our project. Firstly, with regards to data collection, we will look to create a more robust dataset for training and testing. Since there is a lot of manual work that comes with identifying drop-shipped brands in specific product categories and across various websites, building a dataset is quite time-intensive, and we believe that the current version of our dataset is limited in generality due to its sole focus on a specific product category as well as its reliance upon AliExpress for authentic product listings. In the future, we will bolster this dataset by increasing product and category diversity within our dataset and sourcing our authentic data from a larger number of sources to make it more representative of the real-life e-commerce industry. This is especially important as with the number of dropshipped products on the market, and the numerous new product types that are dropshipped, a generalized model will help with the real-world application.

Another area of improvement for our project would be to examine the results of applying a number of additional classification and feature selection algorithms to the task at hand. We see the possibility of a more semantic or syntax-oriented embedding model such as GloVe improving model performance meaningfully, as it would be able to identify errors made in hastily fabricating product descriptions. In addition, Given the current excitement surrounding decoder models, such as GPT models, we feel they could provide interesting results or solutions to the issue. However, we stand that from a theoretical point encoder based models should better understand context. We also would have loved to try further hyperparameter tuning, for example running on more folds for the K-fold cross-validation or trying different learning rates.

Finally, we'd like to look further into decreasing compute costs. As a consumer facing app, we'd ideally like to have the least overhead possible. The performance of ALBERT compared to RoBERTa and BERT showed promise. Further, principal component analysis for the Word2Vec model, being able to reduce down the dimensions and make the model less complicated.

By making these enhancements across data procurement, feature extraction, and model creation, we look to improve the ability for our model to be applied across generic e-commerce use cases. Additionally, we hope that our work can serve as a helpful reference point for others that may be researching similar problems, and that it can be built off of to further learning in the scientific community and product transparency in the world of e-commerce.

9 Acknowledgement

Firstly, we would like to thank Professor Adam Meyers, whose effort in teaching this course on natural language processing provided us with the basis upon which we could complete our final research project. In addition, his guidance and suggestions throughout our entire project were invaluable in helping us transform our ideas into reality. In addition, we would also like to thank our graduate mentor, Sahil Bakshi, for his responsiveness and amazing openness to helping us.

References

[1] Padurariu, C., Breaban, M. E. (2019). Dealing with Data Imbalance in Text Classification. In *Proceedings of the 23rd International Conference on Knowledge-Based and Intelligent Information Engineering Systems* (pp. 736-745). Procedia Computer Science, 159.

[2] 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).

[3] Li, J., Ye, Z., Xiao, L. (2019). Detection of Propaganda Using Logistic Regression. *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda.*

[4] Devlin, J., Chang, M.-W., Lee, K., Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (pp. 4171-4186). Association for Computational Linguistics.

[5] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., Polosukhin, I. (2017). Attention Is All You Need. In *Advances in Neural Information Processing Systems 30 (NIPS 2017).*

[6] Babu, P., Yandrapati Eswari, Rajagopal(2020). Classification of COVID-19 Tweets Using Pre-trained Language Models. In *6th Workshop on Noisy User-generated Text (W-NUT).*