Multi-layer stacking with diverse models to determine quality of Lazada's product titles

Akshay Bhonde
Unilever
Singapore
akshay.bhonde7@gmail.com

Samarth Agarwal
DBS
Singapore
samarthagarwal23@gmail.com

ABSTRACT

As the number of products in E-commerce websites increases multifold to the order of millions, it is imperative to have clean and relevant product titles to ensure positive customer experience. The task of CIKM 2017 Cup was to build a supervised learning model that can automatically grade the clarity and conciseness of a product title. The overall strategy was to generate diverse set of features from product title and category data to cover high level themes like title relevance to category, categorical representation in title, redundancy in titles, and NLP features like word clustering, word sequence and word/character counts. A 3-layer modeling stacking framework [1] was built to generalize by combining output from different models in an efficient way. Maximum gain was achieved by combining boosting/bagging models built on the text and category features and LSTM models generated from word embeddings. All layer 2 models were combined using a single ridge regression model in Layer 3 along with probability calibration through isotonic regression to assign optimum weights and optimized probability estimates. This solution achieves highest overall accuracy in the competition with RMSE of 0.242106 for clarity and 0.32885 for conciseness.

KEYWORDS

Machine learning, Supervised learning, Natural language processing, Ensemble, Stacking, CIKM

ACM Reference format:

Akshay Bhonde and Samarth Agarwal. 2017. Multi-layer stacking with diverse supervised models to determine quality of product titles. In *International Conference on Information and Knowledge Management, Pan Pacific Singapore, 6-10 Nov. 2017 (CIKM'17),* 4 pages.
DOI:

1 INTRODUCTION

CIKM 2017 challenge was to determine the quality of Lazada's product titles using two metrics i.e. clarity and conciseness (labeled by Lazada's internal QC team). The available data included product titles and descriptions, three level category hierarchy, price of the product and location where it was being sold. The approach was to extract information primarily from product titles and category hierarchy. Two separate supervised learning binary models were employed to predict each metric.
Problems with similar context have been solved on Kaggle where product search query are compared to returned product titles to model search relevance [2 – 3]

1.1 Data

The data is divided into 3 parts – *Training data* consists of more than 36 thousand product titles along with clarity and conciseness labels. *Validation data* consists of more than 11 thousand product titles without labels. This data is used for Validation Phases (Phase 1). *Testing data* consists of more than 12 thousand product titles without labels. It is used for the final evaluation (Phase 2).
The evaluation metric is Root-Mean-Squared-Error (RMSE)

2 METHODOLOGY

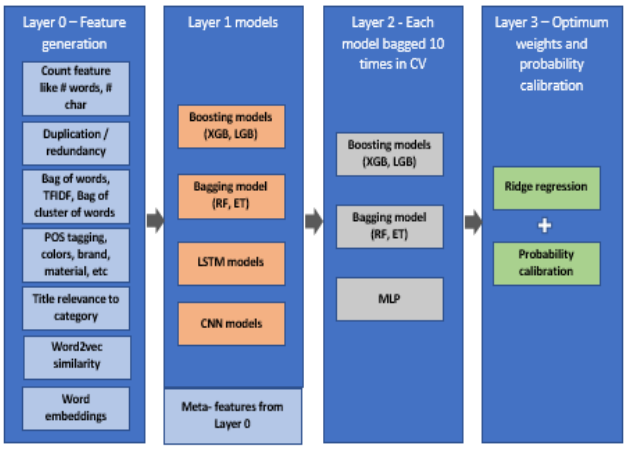

Figure 1: Three-layer modeling architecture diagram to ensure the optimum robust model

The intent was to build a set of diverse features to obtain a competitive score with a single model. Once this was achieved, an ensemble of models was built with varied features in each model. The training set was divided into 5 folds and each model was cross validated using these folds. The different types of models in Level

1 included an Xtreme Gradient Boosting (XGBoost) model, LightGBM, Random Forest Classifier, ExtraTrees Classifier. A state-of-the-art recurrent neural network model viz. the Long-Short-Term Memory model was also built to learn from the sequence of words of the product titles. The product titles were transformed into vectors using two readily available text corpuses – the Google news corpus and GloVe embeddings [4-5]. The model was trained on these word vectors. Finally, some randomness was incorporated in feature selection and parameter selection to create diverse Level 1 models.

2.1 Feature Engineering

2.1.1 Category features -

a. *Label encoding* - Label encoding was opted instead of one-hot due to similar performance and less dimensionality
b. *Bayesian categorical encoding* - These were very significant. We obtained cross-validation priors for each category with some random noise as features to avoid overfitting [6].
c. *PCA of one-hot encoded* features

2.1.2 Duplication/redundancy in title - Features that help identify duplication / redundancy in text proved to be very significant in both the models (more in concise). Duplication was identified in following ways

a. Ratio of unique words in title by number of words in title
b. Ratio of unique n-char grams in title by total number of n-char grams in title (3 to 8 grams were used)
c. Analyzed word2vec [7] similarity of each word in title and created features that gave number of words with similarity greater than one mentioned below
    i. sim_value (Ratios also included)
    ii. sim_value = 0.98
    iii. sim_value = 0.99
    iv. sim_value = 0.995
    v. sim_value = 0.999
    vi. sim_value > 0.98 and sim_value < 0.9995 (To remove the words that always occur together like eau de perfume)

2.1.3 Simple word/char count features – Observed that count feature of number of words, average character per words, counts and ratios of spaces, characters, numbers, extras were significant in both models

2.1.4 Part of speech count features - Nouns and adjectives

2.1.5 Identifying attributes in title - Feature indicating count of color, material, gender, brand, measurement

2.1.6 Word count features -

a. Count-vectorizer – A sparse matrix
b. Cluster words using word2vec embedding to reduce dimensionality of count-vectorizer and form meaningful clusters e.g., gender, measurement, color etc. related attributes

2.1.7 Price of the product – All converted to single currency

2.1.8 Data Cleaning – Multiple spelling mistakes were observed. Two approaches were used -

a. A systematic approach to identify mistakes like 'batterr', 'bluebook', etch was built. All the words were arranged in alphabetical order and for words occurring less than 10 times, edit distance was calculated with neighboring word in alphabetical ordered list. If edit distance was less than 0.2, both the words were stored in a list like the one in Fig. 2. ~1000 such combinations were fixed.
b. Automated correction of certain standard abbreviations like "mm" meaning millimeter, "blk" meaning black, "wht" meaning white, etc. was done to improve feature significance.

2.1.9 Tf-Idf scores – Tf-Idf scores of the words in the title (using each title and description as a document as well as combining all titles and descriptions in distinct category level 3 to treat as a single document)

```
[('abbot', 'abbott'),
 ('abdome', 'abdomen'),
 ('absol', 'absolu'),
 ('absolu', 'absolue'),
 ('absolue', 'absolut'),
 ('absolut', 'absoluta'),
 ('absoluta', 'absolute'),
 ('absorben', 'absorbent'),
 ('absorbtion', 'absorption'),
 ('acaci', 'acacia'),
 ('acceseries', 'accesories'),
 ('accesories', 'accesorios'),
 ('accessorie', 'accessories'),
 ('acryli', 'acrylic'),
```

Figure 2: Spelling corrections on the training set

2.1.10 Word2vec similarity - The Google news corpus was used to build a pre-trained Word2vec model. Similarity scores were then calculated between category and title strings. An important feature was calculating the similarity with each one of the categories and not just the corresponding one. Moreover, highest similarity score between words such as "brand", "material", "quality", "measurements" and each word in the title were also used as features to extract the corresponding information from the product title

2.1.11 Model predictions from other models was also used as features due to correlation between conciseness and clarity.

For concise, the most important features were the length, number of characters, unique words, words with vowels and n-char combinations of characters in the title. Feature selection was done using Recursive Feature Elimination using cross validation and Random Forest's top-n important features (multiple models with variable n).

2.2 Modeling

XGBoost and LightGBM provided almost similar results and proved to be the best single models for both metrics. A simple average of this model with an LSTM model [8] built using the

Google news vectors embedding led to a top 5 score on the leaderboard.

Parameter tuning on the above set of models did not result in much gain on the leaderboard and it was decided to focus on building/testing a multi-layered stacked model framework to combine the predictions of different models (diversity created by randomizing the model algorithm, model parameters and input features). Approximately, 30 models were constructed.

Recursive feature elimination through cross validation was employed on Level 1 model predictions combined with input features to generate a set of Level 2 features. These models were bagged 10 times to avoid overfitting the data whose risk was quite high due to a relatively small training set and model predictions being used as features. This increased total run-time but also the made the models quite stable. A simple average of the predictions of this set of 80 models helped us gain top 3 leaderboard position.

Since, simply averaging Level 2 model predictions did not provide substantial gain in accuracy, the focus shifted towards optimally combining Level 2 model predictions. Ridge and Lasso regression were tried as single models in Level 3 layer to utilize the benefits of regularization. Ridge provided better results compared to Lasso and led to 2nd position on leaderboard.

It was observed that the probability distributions of different L2 predictions varied, especially minimum and maximum probability value. Ridge regression solved the problem of combining different features through optimum weights to an extent. It does not solve the problem of readjusting the probability estimates and since, the metric for the competition is RMSE, having correct probability estimate is important. Probability calibration through Isotonic regression and Platt's scaling [9 – 10] was tried. Isotonic regression provided better results. This was the final submission which led to the first rank on leaderboard.

3 FINAL MODEL DETAILS

3.1 Model parameters for Level 1 models

3.1.1 Xtreme Gradient Boosting Models

1. Learning rate = 0.005
2. Max Tree depth = 6
3. Number of iterations = 4000
4. L1 regularization = 0.9
5. Data subsample = 0.7
6. Column sample by tree = 0.7

3.1.2 Light Gradient Boosting Models

1. Learning rate = 0.001
2. Number of leaves = 50
3. Number of iterations = 1000
4. Feature fraction = 0.7
5. Bagging fraction = 0.7
6. Min data in leaf = 1
7. Bagging frequency = 1

3.1.3 Random Forest and Extra trees classifier

1. Number of estimators = 200
2. Max features = 'sqrt'
3. Max depth = 10
4. Random state = 0

3.1.4 Long-Short-Term Memory Neural Network Models

Embeddings used - Glove 42Bn 300d vector and Google News

Features

1. Basic cleaned titles
2. Titles with more cleaning

Parameters

1. Sequence length = (25,30,35,45)
2. LSTM units = 100
3. Number of neurons in dense layer = 200
4. Dropout = 0.2
5. Recurrent dropout = 0.2
6. Activation - ReLu
7. Optimizer - Adam

3.1.5 Convolution neural network

Combining Conv1D with multiple filter length (2, 3 and 4) using graph. No. of filters = 32. Fully connected to 64 neurons.

3.2 Model parameters for Level 2 models

3.2.1 Xtreme Gradient Boosting Models

1. Learning rate = Random between (0.005,0.01)
2. Max Tree depth = Random (3,4,5,6)
3. Number of iterations = 4000
4. L1 regularization = Random (0.1,0.3,0.6, 0.7,0.9)
5. Min child weight = Random (5, 10, 15, 20)
6. Data subsample = 0.7
7. Column sample by tree = 0.7

3.2.2 Light Gradient Boosting Models

1. Learning rate = Random between (0.005,0.01)
2. Number of leaves = Random (30,40,50,60,70)
3. Number of iterations = 1000
4. Min child weight = Random (5, 10, 15, 20)
5. Feature fraction = 0.7
6. Bagging fraction = 0.7
7. Min data in leaf = 1
8. Bagging frequency = 1

3.2.3 Random Forest and Extra trees classifier

1. Number of estimators = 2000
2. Bagging iterations = 10
3. Max features = 'sqrt'
4. Max depth = 10
5. Random state = 0

3.3 Level 3 Model parameters for clarity

Calibrated Classifier with Ridge Classifier

1. Alpha (L1) = 0.00001
2. Method = Isotonic
3. CV = 5

3.4 Level 3 Model parameters for concise

Ridge Regression

Alpha (L1) = Optimum (0.0001,0.001, 0.01, 0.1, 1.0,5, 10.0,11,15,100, 500)

3.5 Machine specs

1. 16 Core Intel Processor @ 2.4GHz with 100 GB RAM

3.6 Compute time

1. Features generation – 6-8 hours
2. Modeling – 2 days

4 ANALYSIS

4.1 Inaccurate tagging

While analyzing the cases with high mean squared error, in order to come up with new features, it was seen that there are few cases where model makes opposite prediction as compared to the label, due to presence of certain keywords, redundancy of words and length of the title. On examination, the labels seem to be inaccurate.

Table 1: Likely inaccurate labelling of concise metric in training data

Product Title	Concise label	Model prediction
Flip Leather Cver for Samsung Galaxy Note 5 (White)	0	0.99
Kids Sports Digital LED Watched Wrist Watch Alarm Date Rubber Wrist Pink	1	0.01
Bifold Wallet Men's Genuine Leather Brown Credit/ID Card Holder Slim Purse Gift Brown	1	0.01

Table 2: Likely inaccurate labelling of clarity metric in training data

Product Title	Clarity label	Model prediction
Hello Kitty 8000mAh Portable Battery Charger (Pink)	0	0.99

Table 1 and 2 highlight some cases where label seems to be inaccurate. The proposed solution approach can be used to identify such cases and resolve them to further improve the clarity and conciseness tagging.

4.2 Score progressions

XGBoost models turned out to be the best individual models. Ensembling this model with LSTM provided maximum gain in accuracy due to the relatively low correlation ~90% as opposed to 96-98% among the other models. Incremental gains were achieved through the three-layered stacking framework as illustrated in Table 3.

Table 3: CV RMSE score progressions on training set – single base model to three layered stacked models

Level	Model	RMSE Clarity CV	RMSE Concise CV
1	XGBoost (Feature set 1)	0.209904	0.329492
1	XGBoost (Feature set 2)	0.210385	0.324895
1	LSTM	0.215856	0.343209
1	CNN	0.217344	0.358818
2	Average of XGBoost, LSTM and CNN	0.208419	0.31940
3	Ridge	0.20651	**0.311055**
3	Ridge with Probability Calibration	**0.20638**	0.32032

5 CONCLUSION

The proposed approach leverages unique features to identify redundancy in titles by finding word similarities using traditional NLP, state-of-the-art word vector approaches, relevance of title to the category and sequential pattern in title text using recurrent neural networks. Further, the overall stacking framework achieves robustness while building model on a medium-sized dataset. The solution has highest overall accuracy across both metrics, clarity and conciseness. The model provides stable predictions and therefore can be scaled to larger and more varied datasets as well. Hopefully, the insights on features and modeling is useful to all e-commerce websites trying to improve quality of their product titles.

REFERENCES

[1] https://mlwave.com/kaggle-ensembling-guide/
[2] http://blog.kaggle.com/2016/05/18/home-depot-product-search-relevance-winners-interview-1st-place-alex-andreas-nurlan/
[3] https://www.kaggle.com/c/crowdflower-search-relevance
[4] https://github.com/mmihaltz/word2vec-GoogleNews-vectors
[5] https://nlp.stanford.edu/projects/glove/
[6] Daniel Micci-Barecca, A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems
[7] Tomas Mikolov et. al., Efficient estimation of word representations in vector space
[8] Klaus Greff et. al., LSTM: A search space odyssey
[9] http://fastml.com/classifier-calibration-with-platts-scaling-and-isotonic-regression/
[10] Caruana et. al., Predicting good probabilities with supervised learning