# Classification on Word2Vec / Doc2Vec

Approaches comparison and evaluation

Michal Stefanik

# Outline

1. Problem (objected use-cases)
2. Data and preprocessing
3. Approaches
4. Evaluation
5. Results

# 1. Use cases:

1. Score a relevance of a piece of text (document) towards the known categories

- To categorize the unlabelled documents
- To enable soft match in content search systems

2. Recognize some similarity between the trained documents

- Common ground for better content-based future recommenders for customers (e.g. K-NN)

Project home: https://github.com/searchisko/project-classifier-poc

# 2 Dataset and categories

- 50 024 docs (of which 43 478 longer than 10 tokens)
- 137 tokens in average
- Documents can be quite similar with both content and semantics
- Hard to categorize for human observer, relevance to category is fuzzy

**Documents origin:** Product's Stackoverflow questions, internal issues, guidelines, technical documentations, blog posts, comments...

**Preprocessing:**

Gensim default (1. x.lower(), 2. strip_tags, 3. strip_punctuation, 4. strip_multiple_whitespaces, 5. strip_numeric, 6. remove_stopwords, 7. strip_short), with stemming persisted (accuracies cca. + 1.5%)

# 2 Dataset and categories

admin shell command ships esb support connect command idea allow script number commands containers root vagrant centos bin admin available commands change rmi registry port changes rmi registry port management layer existing container instance change rmi server port changes rmi server port management layer existing container instance change ssh port changes secure shell port existing container instance create creates new container instance destroy destroys existing container instance list lists existing container instances start starts existing container instance stop stops existing container instance type command help help specified command

Figure 1: preprocessed document of close-to average size, origin category: fuse

jboss dev studio jboss seam final richfaces want use richfaces components business logic single jar component tool file contain suppose customized data table called application passing parameters sortable skin want work kind api applications includes possible ejb seam project kindly provide reference

Figure 2: preprocessed document, origin category: developer studio

# 2 Dataset and categories

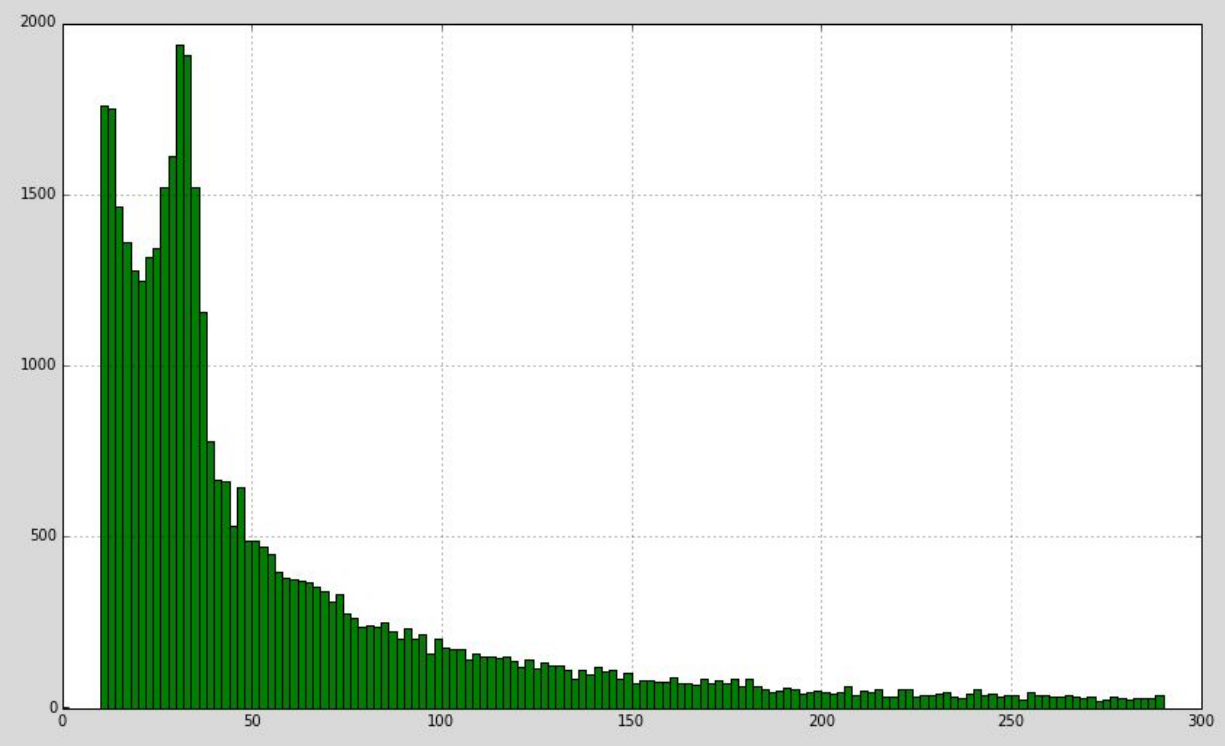



Figure 3: preprocessed training documents distribution by size in no. of tokens

# 2 Dataset and categories

- 14 very unbalanced categories
  - (5 to 18 302 docs, 3105 mean size)
- Each category is a content of one RedHat product
- Categories might have a significant intersection semantically as well as by vocabulary

| | label | accuracy | size |
|---|---|---|---|
| **11** | softwarecollections | 0.000000 | 5 |
| **9** | developertoolset | 0.000000 | 12 |
| **8** | cdk | 0.111429 | 72 |
| **12** | mobileplatform | 0.483810 | 101 |
| **3** | datagrid | 0.417256 | 314 |
| **0** | amq | 0.420341 | 654 |
| **2** | webserver | 0.331566 | 956 |
| **6** | bpmsuite | 0.523567 | 1123 |
| **5** | brms | 0.475673 | 1358 |
| **7** | devstudio | 0.842576 | 3373 |
| **13** | openshift | 0.922969 | 3492 |
| **10** | rhel | 0.955152 | 6511 |
| **4** | fuse | 0.875087 | 7205 |
| **1** | eap | 0.949349 | 18302 |

Figure 4: categories with size and LogReg's classifier accuracies

# 3 Approaches

## 3.1 Word2Vec

- Independent training (sg or cbow) of 14 models on whole wordlist of given cat.
- Models sharing the same pre-init vocabulary, standalone trainings
- Relavance towards categories by gensim_W2V.score() for each category [0]
- Classification to top-scored categories for each doc
- Evaluation of accuracy in CV manner

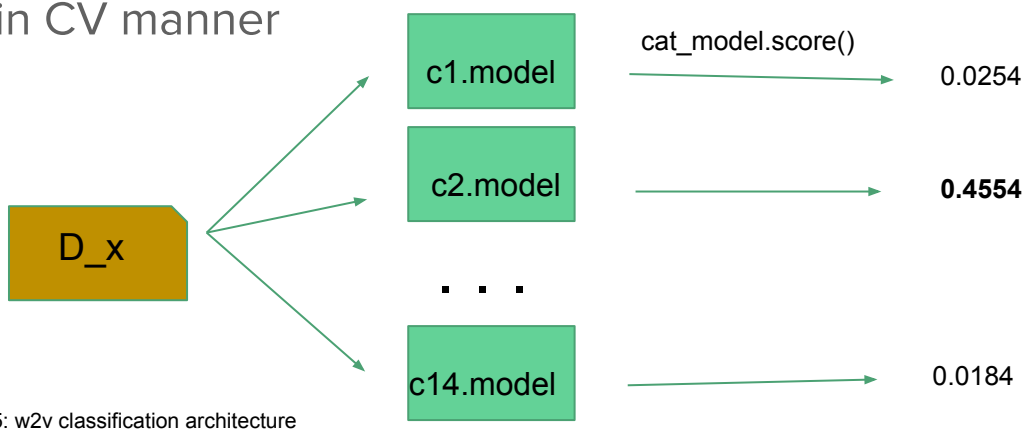



Figure 5: w2v classification architecture

# 3 Approaches

## 3.2 Doc2Vec

- Single doc2vec model repeatedly trained on all documents at once
- Documents shuffled in epochs [1] - accuracy no longer increases after 10+ epochs
- Vectors inference - slight accuracy improvement for 20+ inference epochs
- Classification to categories by selected adjacent classifiers (NN, SVM, LogReg)

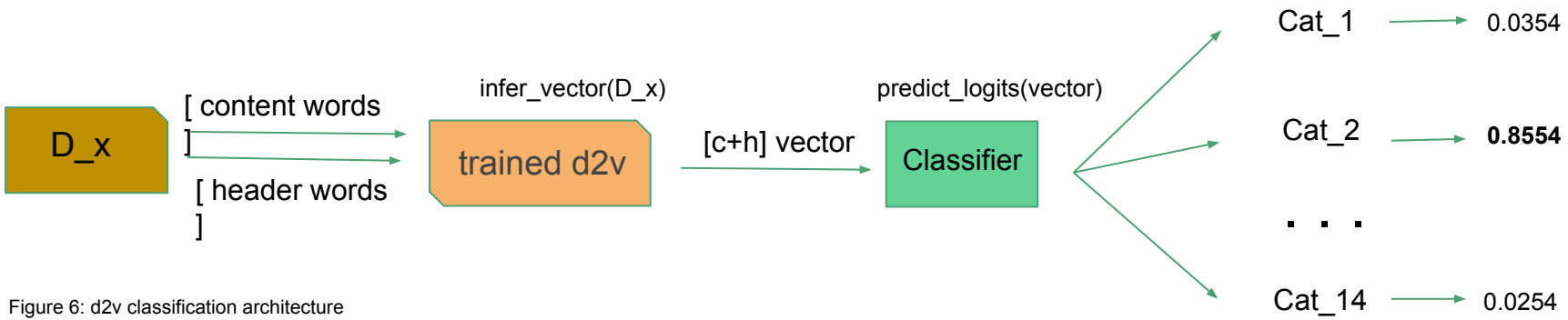
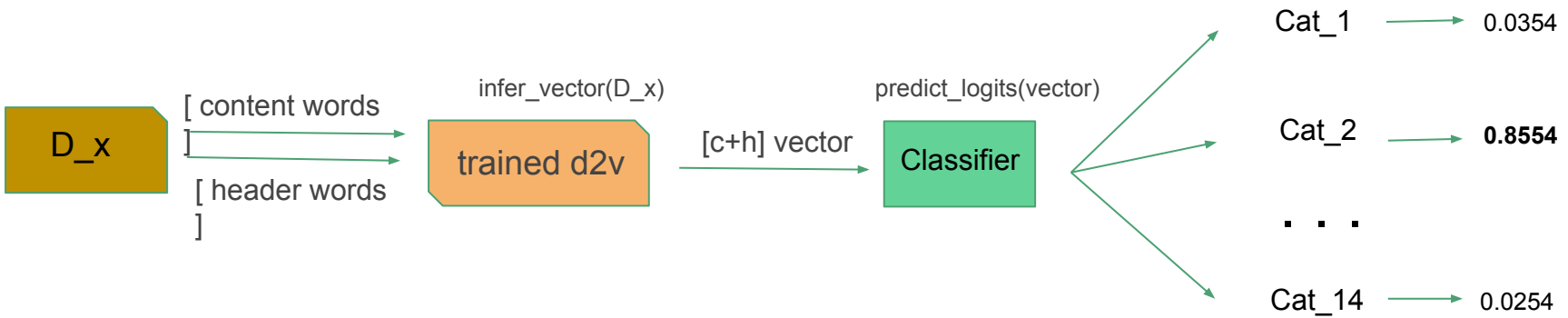



Figure 6: d2v classification architecture

# 3 Approaches

## 3.2 Doc2Vec

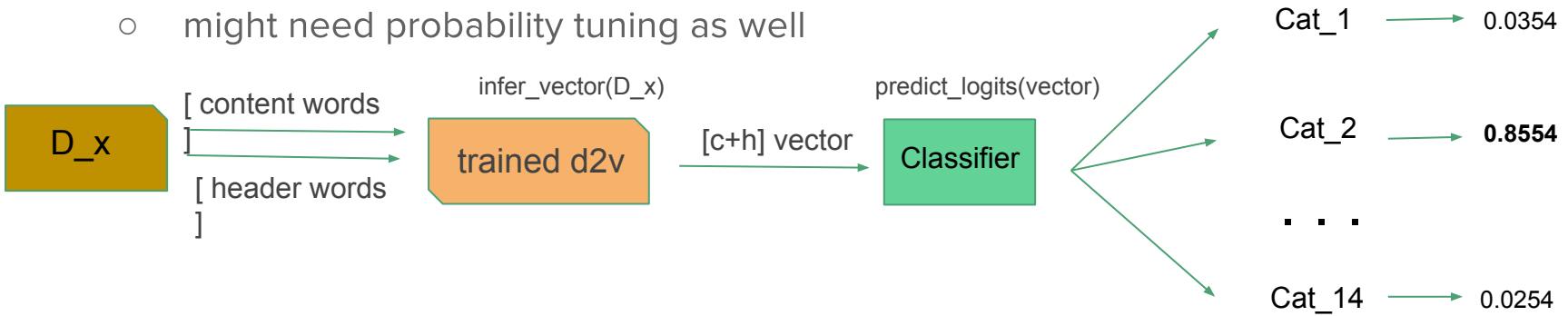
- Hyperparam tuning of d2v brings little accuracy increase (< 1-3% compared to default)
- Changes in [preprocessing, training algo, inference_epochs, training_epochs] affect accuracy similarly for all classifiers
- Classifiers have different optimal length of input vector (e.g. LogReg cca. 800+800, NN cca 500+500)
    - Stil variations from 300 vectors onward is quite small: < 2%



# 3.2 Doc2Vec

## 3.2.1 Adjacent classifiers

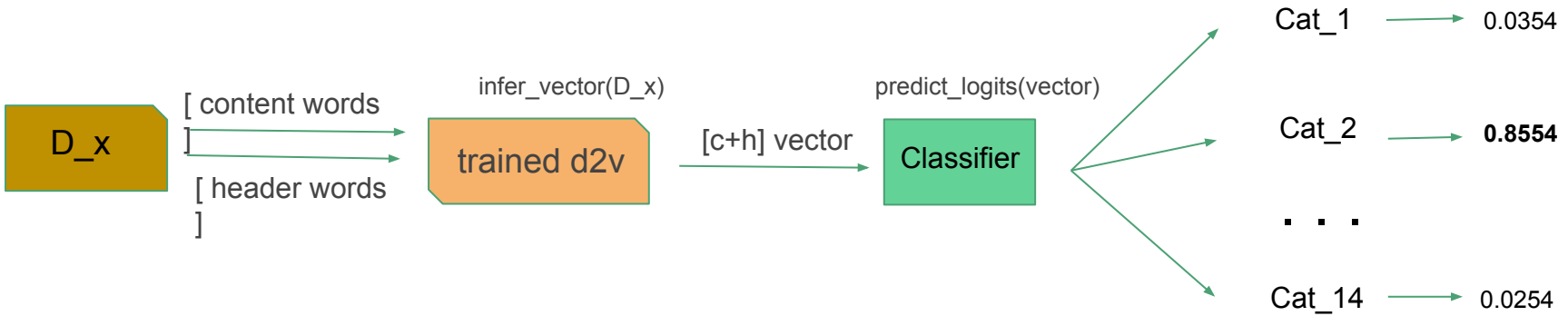
- SVM:
  - best linear and radial kernel
  - grid searched C and gamma params (C=0.1 for linear kernel, C=8 for radial)
  - radial kernel brings accuracy increase of only cca 0.3%
  - sklearn implementations
  - accuracy comparable to LogReg: 14 categories on 83.88% (radial), 83.59% (linear)
  - not evaluated with headers
  - might need probability tuning as well



# 3.2 Doc2Vec

## 3.2.1 Adjacent classifiers

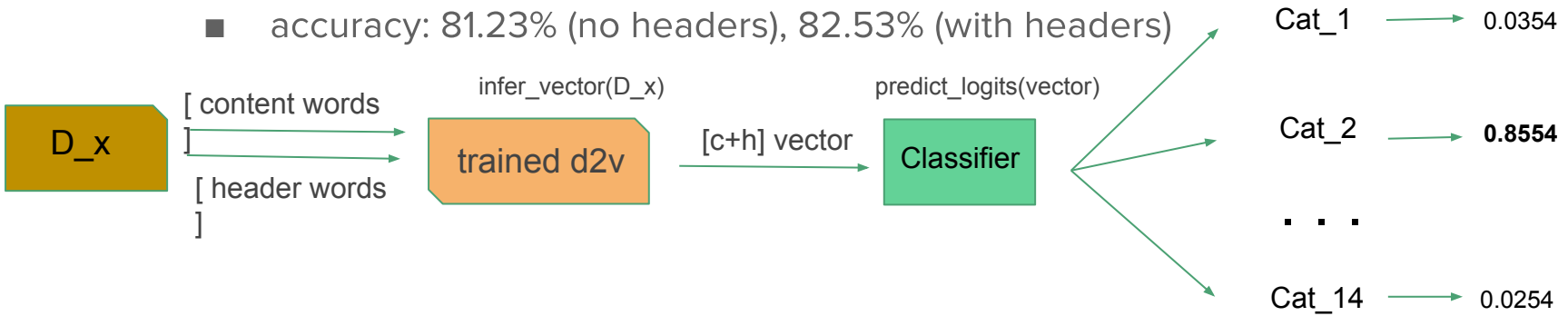
- Logistic Regression:
  - grid searched C (optimal C=0.3)
  - similarly fast as Linear SVM (40k docs in 1000vectors in cca 5min)
  - sklearn implementation
  - similar accuracy to SVM: 14 categories on 83.12% (v500 no headers), or
  - 87.01% (v800, headers)
  - easy and behaving scoring using native logits



# 3.2 Doc2Vec

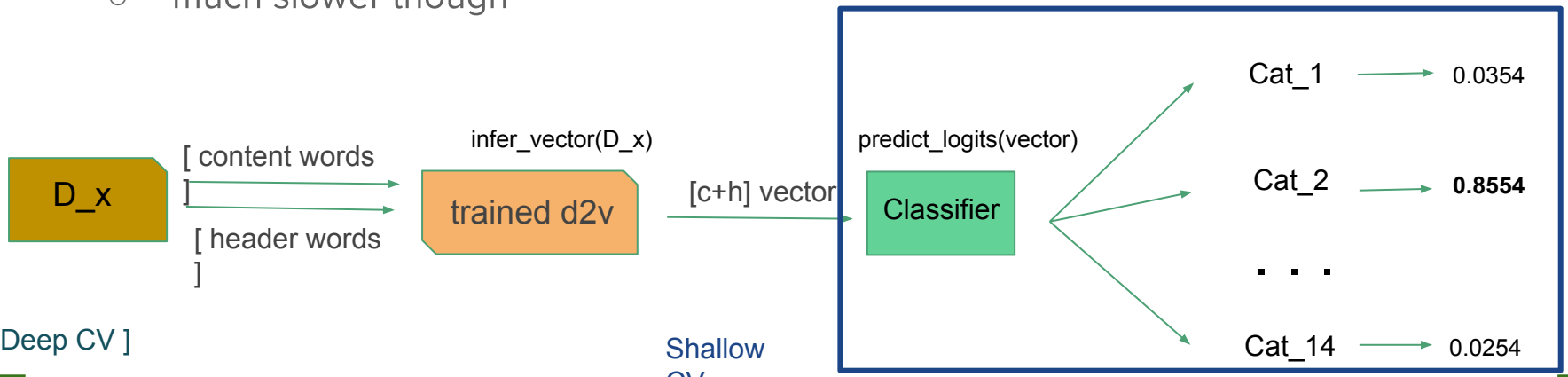
## 3.2.1 Adjacent classifiers

- Neural Network [1]
  - built with Tensorflow 1.0
  - CPU (no GPU) only, though still faster (2-3x) than all others
  - Architecture inspired by [1]: one fully-connected hidden layer of size [vectors/2]
  - best candidate for biggest 3 categories: accuracy close to 95% (v500, no headers)
    - Better by cca 1.5% than SVM, LogReg
  - but not so good on the whole unbalanced category set:
    - accuracy: 81.23% (no headers), 82.53% (with headers)



# 3.2.2 Evaluation strategies

- Shallow:
  - Training of d2v on whole dataset, d2v used simply as transformer of documents to vectors
  - only train/test of adjacent classifier included in CV cycle
- Deep
  - Train/test of both d2v and adjacent classifier included in CV cycle
  - Yields very similar results to shallow (5-fold CV, LogReg on C=0.3 gives cca. **-0.3%** in accuracy)
  - much slower though



# 3.3 Common observations

- Default window size (8) worked best (at 10 and 6 accuracy decrease cca. 2%)
- gensim w2v's skip-gram gives very little improvement in score() accuracy if any, but was 11x slower [figure 9], though claimed to work better for small categories [2]
- w2v's score() behave unusably for too little categories' size [figure 12]
    - but might be still good for big enough categories
- d2v accuracy si perhaps independent on deep/shallow training strategy
- Gensim's default preprocessing is not too fast - it was a bottleneck while gathering the content from web APIs
- Gensim's default parameters were often the best config, or close to optimum (e.g. window=8)

# 4.1 Evaluation

## Word2Vec:

- Accuracy of the single-target classification based on a selection of top-scored category for each doc

## Doc2Vec:

- Accuracy of the classification based on logit/softmax/class_probs top-scored category for each doc (LogReg/NN/SVM)

# 4.2 Results

Accuracy increasing with category size:

| | label | accuracy | size |
|---|---|---|---|
| **11** | softwarecollections | 0.000000 | 5 |
| **9** | developertoolset | 0.333333 | 12 |
| **8** | cdk | 0.100000 | 72 |
| **12** | mobileplatform | 0.682143 | 101 |
| **3** | datagrid | 0.301587 | 314 |
| **0** | amq | 0.141221 | 654 |
| **2** | webserver | 0.279355 | 956 |
| **6** | bpmsuite | 0.351111 | 1123 |
| **5** | brms | 0.318015 | 1358 |
| **7** | devstudio | 0.662963 | 3373 |
| **13** | openshift | 0.731044 | 3492 |
| **10** | rhel | 0.890979 | 6511 |
| **4** | fuse | 0.700902 | 7205 |
| **1** | eap | 0.863425 | 18302 |

Figure 7: accuracies of w2v for all 14 categories

| | label | accuracy | size |
|---|---|---|---|
| **11** | softwarecollections | 0.000000 | 5 |
| **9** | developertoolset | 0.000000 | 12 |
| **8** | cdk | 0.111429 | 72 |
| **12** | mobileplatform | 0.483810 | 101 |
| **3** | datagrid | 0.417256 | 314 |
| **0** | amq | 0.420341 | 654 |
| **2** | webserver | 0.331566 | 956 |
| **6** | bpmsuite | 0.523567 | 1123 |
| **5** | brms | 0.475673 | 1358 |
| **7** | devstudio | 0.842576 | 3373 |
| **13** | openshift | 0.922969 | 3492 |
| **10** | rhel | 0.955152 | 6511 |
| **4** | fuse | 0.875087 | 7205 |
| **1** | eap | 0.949349 | 18302 |

Figure 8: accuracies of d2v with LogReg

# 4.2 Results

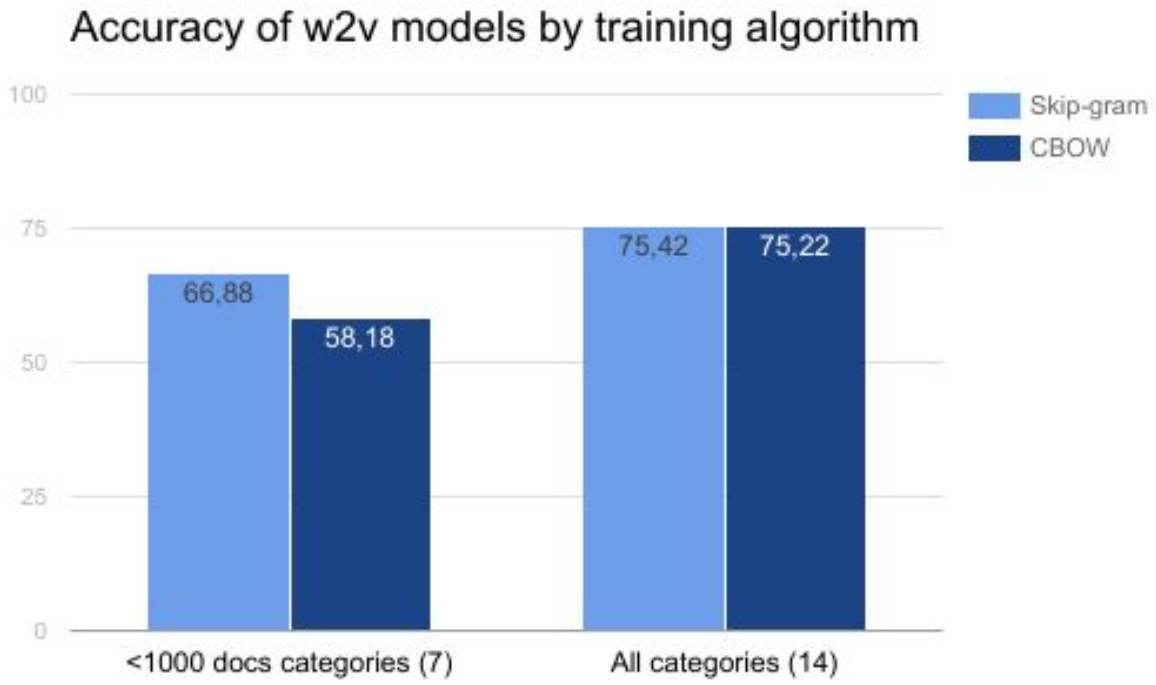



figure 9: Skip-gram vs. Continuous Bag of Words training: accuracies

# 4.2 Results

Error rate relative to document length (in tokens) as classified by Logistic Regression

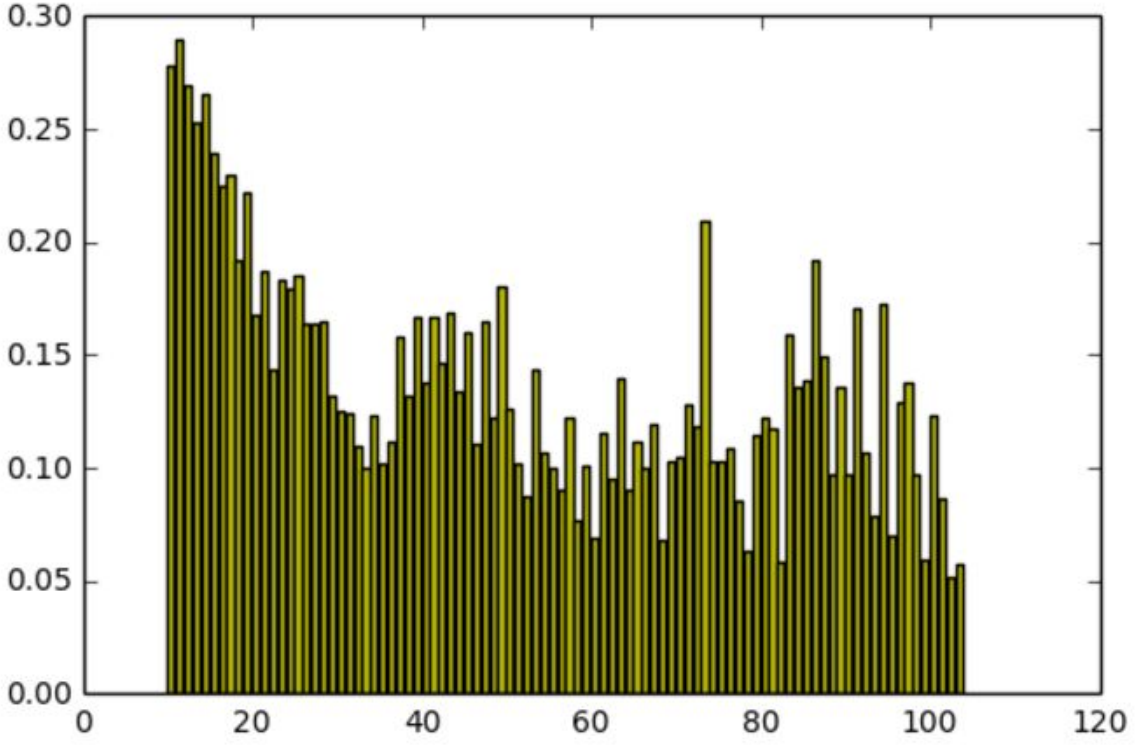



Figure 10: Logreg missclassification rate

# 4.2 Results

Error rate relative to document length (in tokens) as classified by Logistic Regression

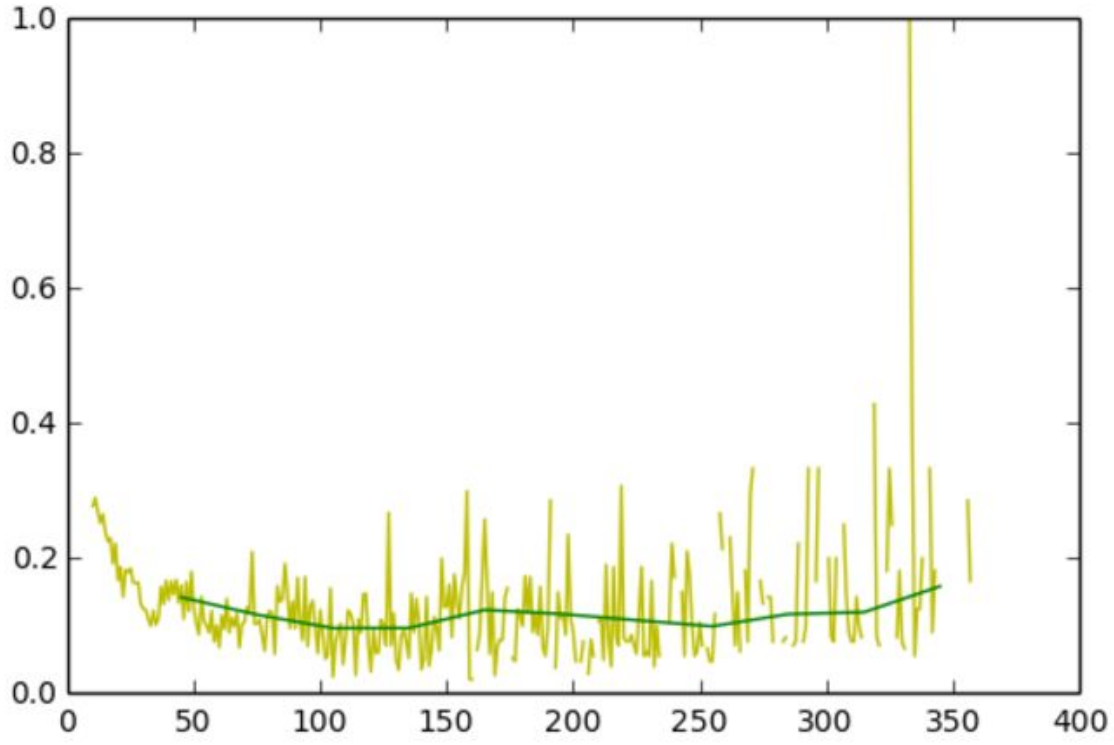
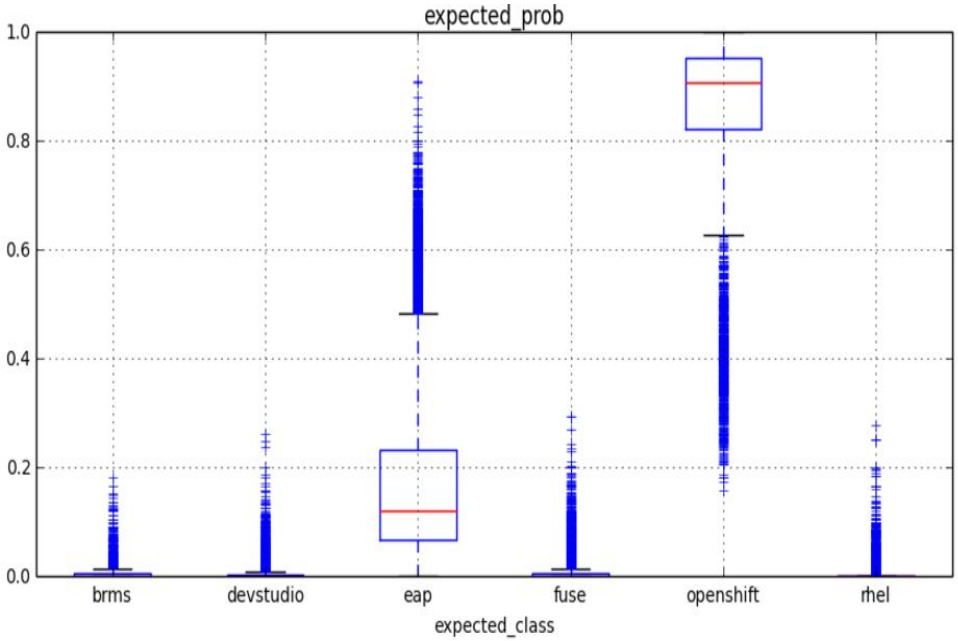



Figure 11: Logreg missclassification rate

# 4.2 Results

[Figure 12]: Scoring of docs towards its original category,

- by v2vec's score (left)
- LogReg's logits on d2v docs vectors (right)



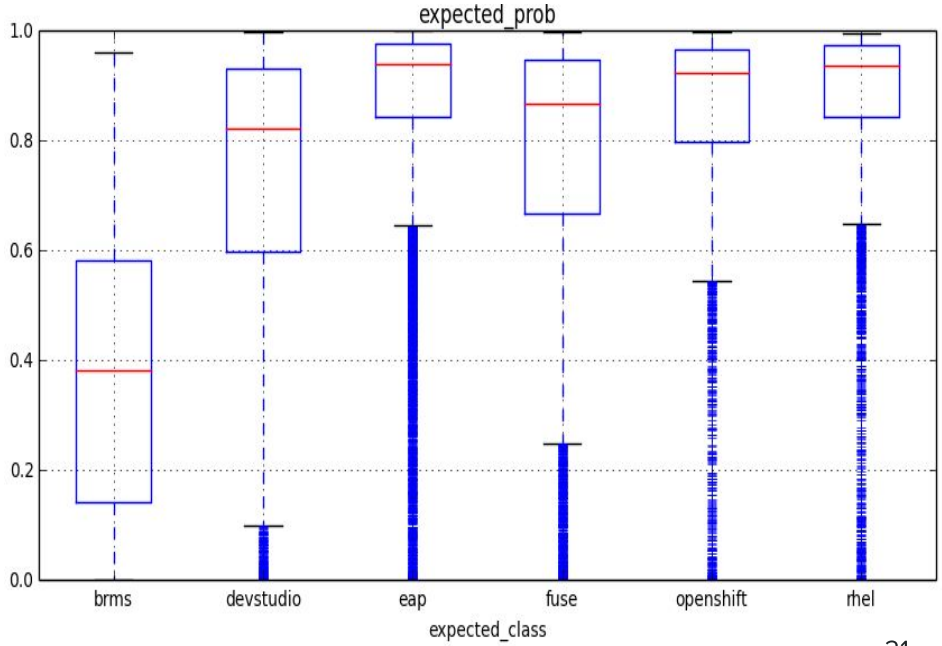




Figure 12: Logits distribution for own category

# 4.2 Results

Overall accuracies of models for categories of various sizes

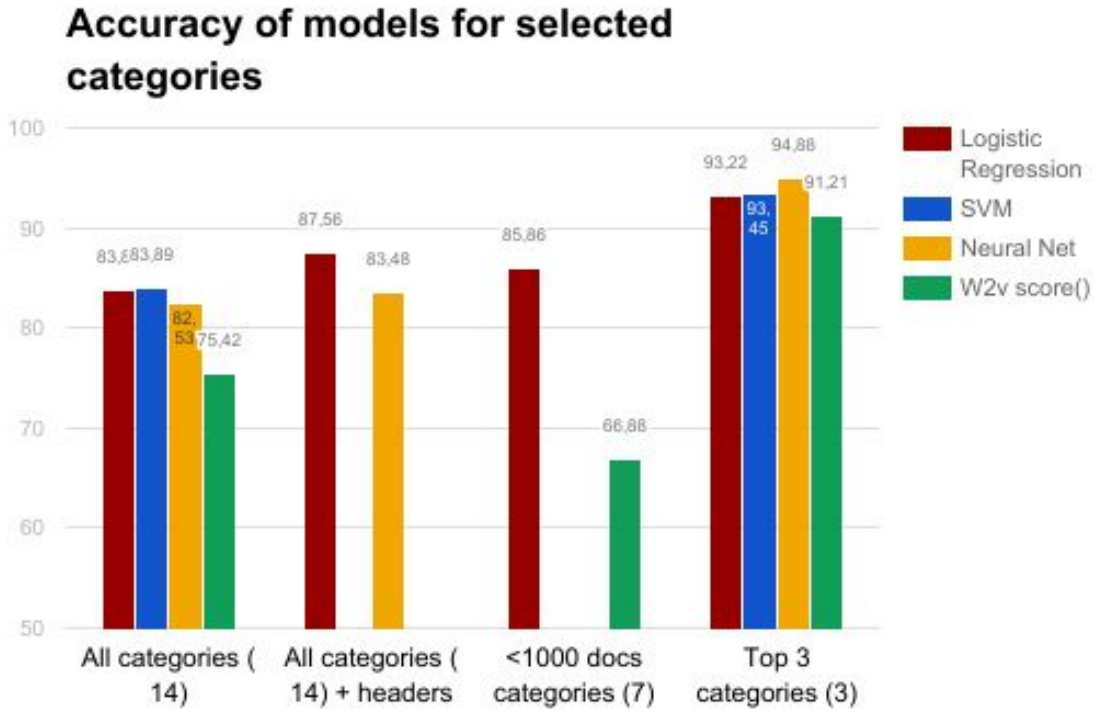



Figure 13: Overall models accuracy

# Thank you!

Michal Štefánik

References:

[0]: https://arxiv.org/pdf/1504.07295v3.pdf
[1 ]: https://deeplearning4j.org/welldressed-recommendation-engine
[2]: http://bit.ly/2p5zV48
[3]: https://github.com/searchisko/project-classifier-poc