Detecting Duplicate Stack Overflow Questions Exploiting the Textual Information, and a Semantic-based Tag Hierarchy

Author: Cristian-Alexandru Botocan - c.a.botocan@student.tudelft.nl

Supervisors: Dr. Maliheh Izadi, Prof. Dr. Arie van Deursen

TU Delft

1. Introduction

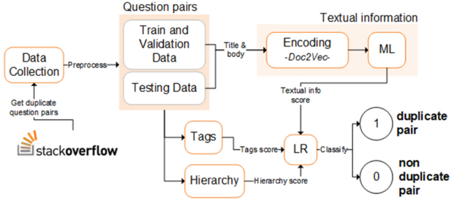
- **Stack Overflow** (SO) is one of the most important online platforms where users can ask questions regarding Software Engineering (SE) topics
- Detecting duplicate SO posts is a manual process done by the maintainers and high reputation users
- Automatic solution increase the efficiency in terms of time and work

2. Research Question

Given a SO question pair (q_i, q_j), where $q_i = (title_i, body_i, tags_i = \{t_1, t_2, \ldots, t_l\})$, we have to assign a label to each pair so that:

- label 1 - duplicate
- label 0 - non-duplicate

3. Approach

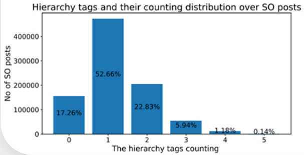


4. Dataset

- Collected using **StackOverflowAPI** on 24.05.2022

Original Dataset

- 688,937 pairs and 897,592 questions



Experiment Dataset

- randomly extract 10.000 pairs

Dataset	#positive instances	#total question pairs	#questions
Training	8,000	16,000	14,627
Validation	1,000	2,000	1,914
Testing	1,000	2,000	1,917

- To avoid bias, generate negative samples with 1:1 ratio for each dataset

5. Textual information score

1. Preprocess title and body

- Extract from body the code snippets and strong components
  - strong annotation: ..., ..., <code>...</code>
  - code snippet annotations: <pre><code>...</code></pre>
- The rest of operations: extract HTML tags, apply mapping rules, word tokenize, stemming

2. Doc2Vec model for title and body embeddings

Hyperparameters	Values Title Embedding	Values Body Embedding
Model	Distributed Memory + Skip-Gram for word vectors	Distributed Bag of Words
Dimension Embeddings	70	600

3. Classify the Embeddings

- Using the ML-based models: *Gaussian NB, DT, KNN*, **SVM**, **LR**

6. Tags score

- Jaccard similarity

$$Score_{tags}(q_i, q_j) = \frac{|tags_{q_i} \cap tags_{q_j}|}{|tags_{q_i} \cup tags_{q_j}|}$$

7. Hierarchy score

Type	Levels	Details
h_mod	3	**Modularity** [2, 3] applied on the SED-KGraph [1] + manually adjusted
h_stat	5	**Statistical Inference** [4, 5] applied on the SED-KGraph [1] + manually adjusted
h_manual	7	**Manually** created based on the h_mod and h_stat
h_full	64	**Automatically** created based on the dendrogram from Agglomerative Clustering

8. Results

Configurations	Accuracy	Recall	F1-score	Precision	Coefficients
Gaussian NB	52.35%	52.73%	52.58%	45.30%	
DT Classifier text	53.40%	53.09%	53.51%	58.30%	
KNN Classifier text	56.95%	54.26%	61.70%	88.50%	
SVM text	83.00%	77.59%	83.16%	92.80%	
SVM text + tags	88.44%	84.05%	88.49%	94.90%	[13.2, 6.34]
SVM text + h_mod	85.25%	80.41%	85.34%	93.20%	[14.2, 3.29]
SVM text + h_mod + tags	88.85%	84.41%	88.89%	**95.30%**	[13.11, 1.63, 5.67]
SVM text + h_stat	87.45%	83.22%	87.50%	93.80%	[13.55, 6.87]
SVM text + h_stat + tags	**89.50%**	**85.45%**	**89.53%**	95.19%	[12.95, 4.24, 4.55]
SVM text + h_manual	83.85%	78.61%	83.98%	93.00%	[14.56, 2.07]
SVM text + h_manual + tags	88.35%	83.90%	88.40%	94.90%	[13.2, 0.54, 6.21]
SVM text + h_full	83.10%	77.72%	83.26%	92.80%	[14.77, 0.91]
SVM text + h_full + tags	88.35%	83.90%	88.40%	94.90%	[13.22, 0.37, 6.32]
LR text	54.55%	54.29%	54.58%	57.49%	
LR text + tags	92.00%	91.66%	92.00%	92.40%	[4.49, 16.04]
LR text + h_mod	76.09%	71.25%	76.41%	87.50%	[4.61, 6.24]
LR text + h_mod + tags	92.00%	91.17%	92.00%	93.00%	[4.48, 1.47, 14.63]
LR text + h_stat	88.30%	83.47%	88.36%	**95.50%**	[4.48, 13.74]
LR text + h_stat + tags	92.05%	90.31%	92.05%	94.19%	[4.49, 4.94, 11.95]
LR text + h_manual	65.40%	63.02%	65.68%	74.50%	[4.65, 4.75]
LR text + h_manual + tags	**92.10%**	**91.68%**	**92.10%**	92.60%	[4.49, 0.23, 15.89]
LR text + h_full	53.90%	53.65%	53.95%	57.20%	[4.73, 0.43]
LR text + h_full + tags	92.00%	91.66%	92.00%	92.40%	[4.47, -0.87, 16.16]

9. Conclusion and Future Work

- Best Configuration vs Best Baseline: +61.72% accuracy, +68.71% recall, +49.27% F1-score
- LR text + h_manual + tags vs LR text (best increasing ratio): +68.83% accuracy, +68.87% recall
- Small hierarchies does not increase the score (h_mod)
- Deep hierarchies does not increase the score (h_full)
- Overall, a hierarchy between 5-10 levels, improves the scores (h_stat, h_manual)

- Explore more on the encoding part: TF-IDF, average Word2Vec, Transformers
- Explore more on the textual information models, use Deep Learning
- Create a specific hierarchy for SO tags, not adapting a Github one
- Take into consideration also code snippets and strong annotations

References

[1] M. Izadi, M. Nejati, and A. Heydarnoori, "Semantically-enhanced topic recommendation system for software projects," arXiv preprint arXiv:2206.00085, 2022

[2] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," Journal of statistical mechanics: theory and experiment, vol. 2008, no. 10, p. P10008, 2008

[3] R. Lambiotte, J.-C. Delvenne, and M. Barahona, "Laplacian dynamics and multiscale modular structure in networks," arXiv preprint arXiv:0812.1770, 2008

[4] T. P. Peixoto, "Bayesian stochastic blockmodeling," Advances in network clustering and blockmodeling, pp. 289–332, 2019

[5] L. Zhang and T. P. Peixoto, "Statistical inference of assortative community structures", Physical Review Research, vol. 2, no. 4, p. 043271, 2020