KU LEUVEN

A comparison of traditional and transformer-based machine learning techniques for NACEBEL classification of Flemish company websites

Gil Coopmans
Ferre Dockx

Promotor: Prof. Dr. Bart Baesens
Daily supervisor: Manon Reusens
Statistics Flanders contact: Jens Van de Weygaert

STATISTIEK VLAANDEREN

Problem statement

NACE codes → NACE**BEL** codes

- Hierarchical structure

NACE code	Type
9XXXX	Service
96XXX	Other services
960XX	Other services
9602X	Hair and beauty
96021	Hairdressers

- Useful for:
  - VAT
  - Sector statistics
  - Government support
  - …

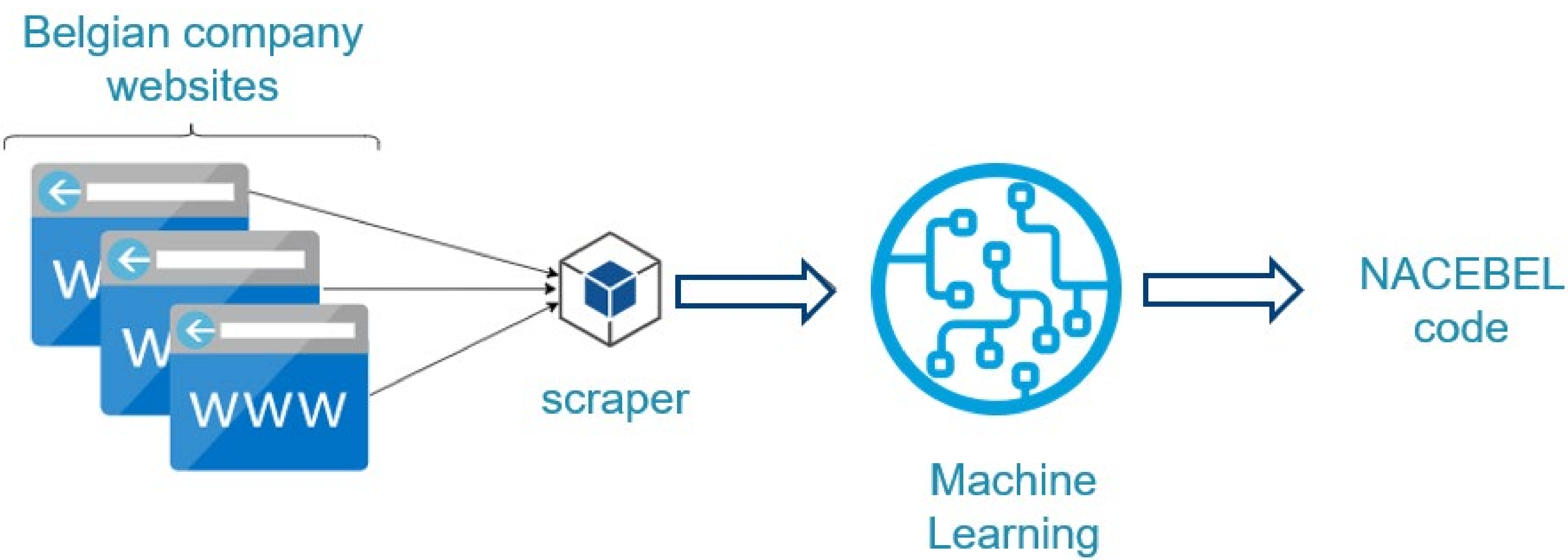
Correct classification is important!

But often wrongly classified

- Human error
- Ambiguitity
- Changes in business activity

KU LEUVEN

Can Machine Learning help?



Literature review

KU LEUVEN

Literature

Predicting NACE codes – previous research

Industry classification based on texts from Dutch company websites (Sinke & Vanthienen (2019))

- Comparing NLP techniques for text classification
- Feature extraction techniques and different models

→ .nl

Exploring a knowledge-based approach to predicting NACE codes of enterprises based on web page texts (Kühnemann et al. (2020))

- SVM and Naïve Bayes
- Improve predictive accuracy with knowledge-based features

→ .nl

Decisions to be made

1. What models to use? (ML vs DL)
2. What data to collect? (HTML/JavaScript, homepage, etc.)
3. How to clean data? (e.g. what to keep)
4. How to pre-process data? (what will we use as input for the models)
5. How to train models? (e.g. how long?, what to test?)
6. How to evaluate the results?

KU LEUVEN

Methodology

Models

Traditional ML method:

- **Logistic regression**
  - ‘Simple’ algorithm
  - Computationally efficient
  - Feature engineering needed
  - Estimates probability of belonging to class

Deep Learning method:

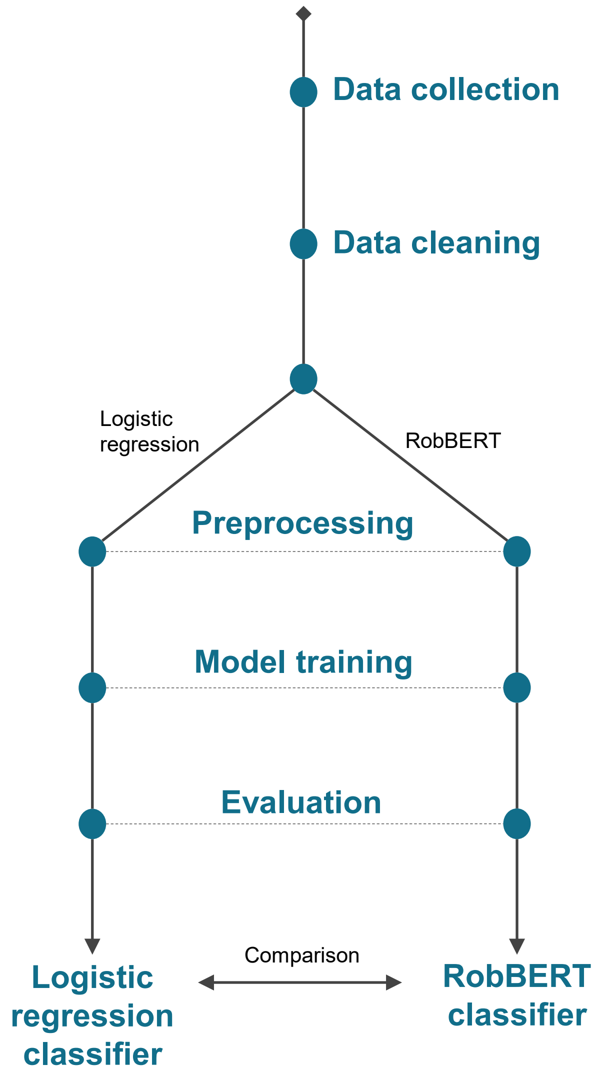
- **RobBERT** (a pre-trained transformer-based Large Language Model)
  - Complex
  - Computationally expensive
  - Neural Network (transformer-based)
  - Automatic feature extraction
  - Fine-tuning

Research questions

"How does a transformer-based model perform on a high-dimensional multi-class text classification task, compared to a traditional machine learning method?"

"What is the classification performance of a transformer-based model compared to a traditional model on different hierarchy levels of NACEBEL codes?"

KU LEUVEN


Data collection
Data cleaning
Logistic regression
RobBERT
Preprocessing
Model training
Evaluation
Comparison
Logistic regression classifier
RobBERT classifier

KU LEUVEN

Data collection

Web scraping

Scraping

STATISTIEK VLAANDEREN

360,000 URLs in dataset**

- Only websites available in Dutch
- Only if scraping is allowed
- Only if the URL is still valid
- Only HTML of homepage is scraped

152,302 scraped web pages (raw HTML)

Linking NACEBEL codes

- 27,5% of companies not in StatBel provided Dataset

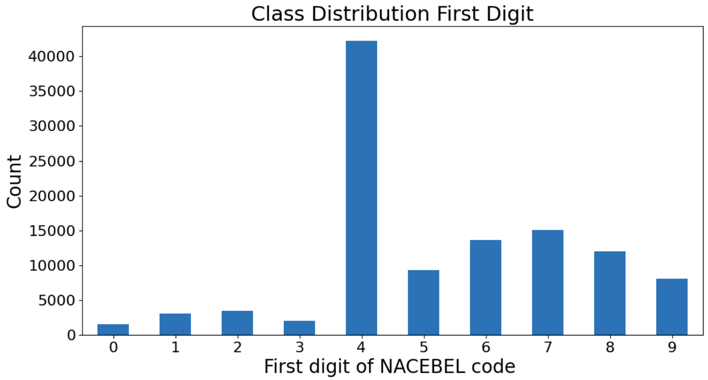
STATBEL België in cijfers

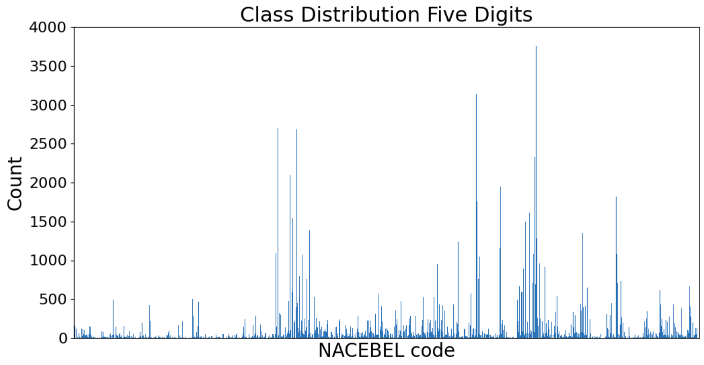
110,372 scraped web pages with NACEBEL code

****The quality of the URL dataset is questionable! → Comparative insights should still be usable**

Data collection

Class imbalance



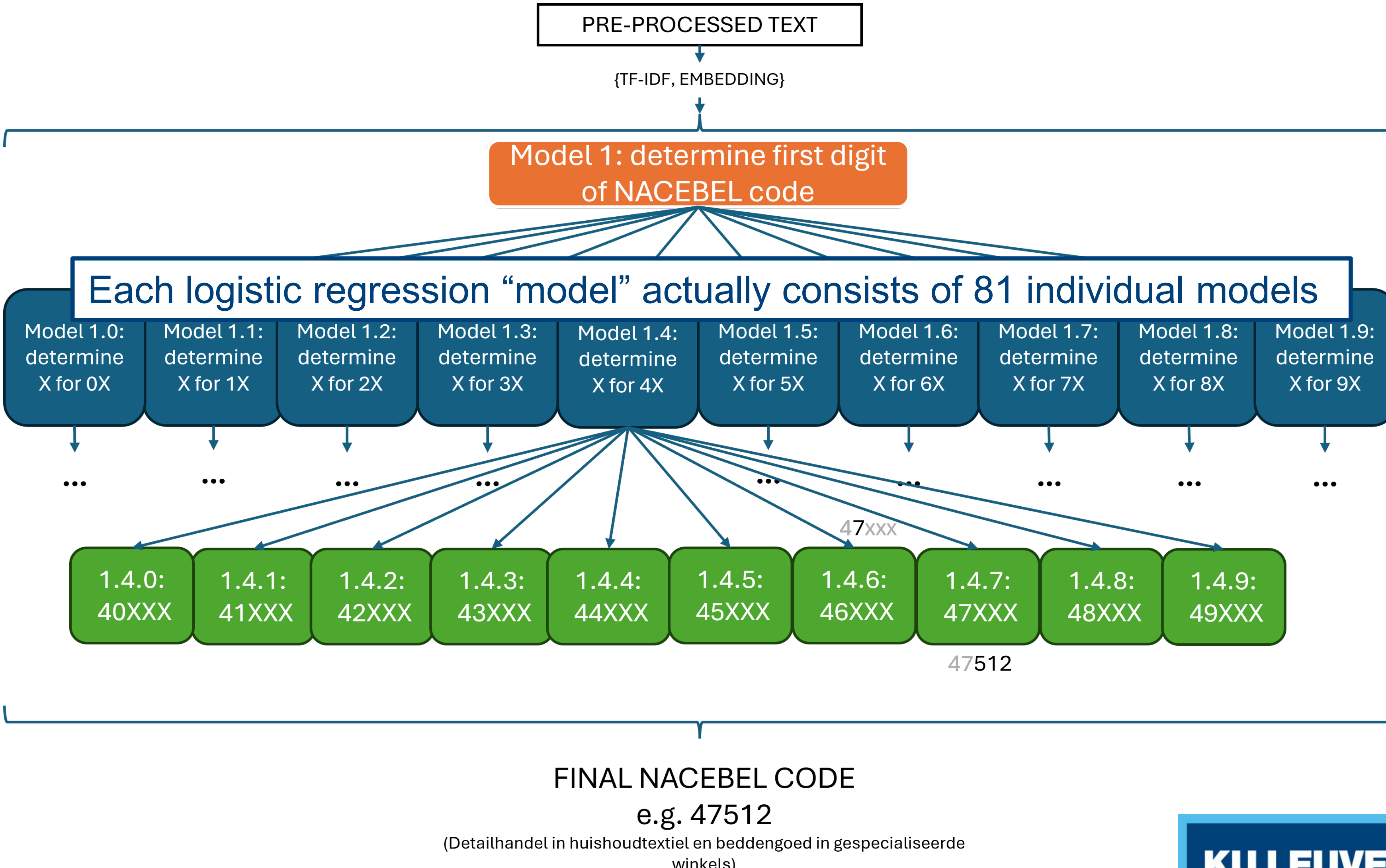


Data cleaning

- Downcasing
- Removing numbers & special characters
- Remove (nearly) empty texts
- Handling duplicates in data
  - Online directories (e.g. ‘data.be’, ‘goudengids.be’)
  - Branches (e.g. McDonald’s)
  - Mistakes URL dataset

  → Remove all but one

Applying logistic regression hierarchically



Logistic regression

PRE-PROCESSING	
Stopwords	Remove ⇔ Keep
Tokens	Full words ⇔ Lemmatization ⇔ Stemming ⇔ Character n-grams
Feature extraction technique	TF-IDF ⇔ Word embeddings
Class balancing	Downsampling on first digit ⇔ Class-Weighting ⇔ Neither

→ 38 experiments with logistic regression

RobBERT

Cleaned data as input

CLEANED TEXT

Tokenizer for pre-trained model

RobBERT Tokenizer

Feed tokenized data to pre-trained language model

Output of RobBERT will be passed to custom classification neural network

Pre-Trained RobBERT model

Custom classification head

OUTPUT LAYER (size = number of unique NACEBEL codes in data)

FINAL NACEBEL CODE
e.g. 47512
(Detailhandel in huishoudtextiel en beddengoed in gespecialiseerde winkels)

Final layer is the classification output
e.g. 47512

RobBERT

- Experiment with different setups
  - Number of hidden layers
  - Size

  …

KU LEUVEN

Evaluation

Accuracy	Weighted F1 score
Can be influenced by majority categories	Better for imbalanced dataset

Results

Results Logistic Regression

	preprocessing-Technique	Feature Extraction	Down-Sample	Class-Weighting	Final Accuracy	Final F1
HEURISTIC (dataset 2F) - stop words kept						
27	STEM	TF-IDF	NO	NO	**0.3783**	**0.3359**
28	STEM	TF-IDF	NO	YES	**0.3673**	**0.3577**

Selected benchmark

→ Duplicate heuristic

→ Keeping stopwords

→ Stemming

→ TF-IDF

→ Class-Weighting

Results RobBERT

ID	Batch Normalization	Layers	Layer Size	Freezing Layers	Down Sampling	Final Accuracy	Final F1
R1	NO	1	768	NO	NO	**0.4356**	**0.3911**
R2	YES	1	4096	NO	NO	**0.4446**	**0.4163**
R3	YES	1	4096	YES	NO	**0.4535**	**0.4187**
R4	YES	1	4096	YES	MED	**0.2680**	**0.2292**
R5	YES	1	4096	YES	2MED	**0.3273**	**0.2934**
R6	YES	2	4096	NO	NO	**0.4322**	**0.4037**
R7	YES	2	4096	YES	NO	**0.4420**	**0.4083**

Selected model

→ One hidden layer

→ 4096 nodes

→ No downsampling

KU LEUVEN

Logistic regression vs RobBERT
Full NACEBEL code

	FINAL ACCURACY
Log reg (28)	0.3673
RobBERT (R3)	0.4535

	FINAL F1
Log reg (28)	0.3577
RobBERT (R3)	0.4187

Logistic regression vs RobBERT

Per digit breakdown

	ACCURACY - 1st DIGIT	ACCURACY - 2 DIGITS	ACCURACY - 3 DIGITS	ACCURACY - 4 DIGITS	FINAL ACCURACY
Log reg (28)	0.6285	0.5299	0.4538	0.4064	0.3673
RobBERT (R3)	0.7117	0.6131	0.5438	0.4928	0.4535
	F1 - 1st digit	F1 - 2 DIGITS	F1 - 3 DIGITS	F1 - 4 DIGITS	FINAL F1
Log reg (28)	0.6417	0.5331	0.4512	0.3999	0.3577
RobBERT (R3)	0.7095	0.6034	0.5223	0.4645	0.4187

- RobBERT outperforms logistic regression
  - All levels
  - All metrics
- Accuracy and F1-score gap increases for RobBERT
  → Hierarchical implementation advantage of logistic regression

- RobBERT still has more room for improvement

Training effort

Logistic regression

- Between 16 and 60 minutes per experiment (19m for benchmark)
- More pre-processing required
- Trained on Google Colab (cpu)

RobBERT

- Between 2 and 4 hours per experiment
- Trained on Google Colab T4 GPU

→ GPU is more expensive

KU LEUVEN

Conclusion

- RobBERT
  - Best performing
  - Room for improvement
    - Adressing class imbalance
    - Use hierarchy of NACEBEL codes
- Logistic regression
  - Shorter training time
  - Less room for improvement

→ Improve data quality & research deep learning applications further

Questions?