Text Classification for International Standardized Codes

Nina Niederhametner
Nina.Niederhametner@statistik.gv.at

Johannes Gussenbauer
Johannes.Gussenbauer@statistik.gv.at

STATISTIK AUSTRIA
Die Informationsmanager

Outline

- Early Attempts
- Neural Network-Based Approaches
- Results
- Deployment
- Conclusions and Outlook

Early Attempts

ISCO Classification (first two digits)
~2014

- **Bag of characters**

- Two Models:
  - **Model 1**: ISCO(2D) ~ Vocabulary
  - **Model 2**: ISCO(2D) ~ Vocabulary + (Sex, Age, NACE-Code, State, Company Size, Social Status)

- **Support Vector Machines** (linear and radial kernel)

ISCO Classification (first two digits)

- **Model 1**

  Accuracy (true positive/total)

	linear kernel	radial kernel
Training-Set	0.8534	0.9269
Test-Set	0.6226	0.6308

Quelle: Verdienststrukturerhebung 2014 (unveröffentliche Ergebnisse)

- **Model 2**

  Accuracy (true positive/total)

	linear kernel	radial kernel
Training-Set	0.7996	0.9992
Test-Set	0.6324	0.4183

Quelle: Verdienststrukturerhebung 2014 (unveröffentliche Ergebnisse)

Neural Network Approach

Currently working on:

Code	Number of Classes	# Instances	Additional Variables
ÖKLAP (custom COICOP for Austria)	>500	Increasing during data collection	Checkbox
ISCO	420	~400.000	Age, Education, NACE2, Citizenship, management role, employment type
ISCED	100	~27.500	Age, Citizenship, ISCO, employment type
NACE	701	~13.000	Age, Education, Citizenship, ISCO2, NUTS-2

Methodological approach
Overview

Input String

Phase 1

String matching dictionary

Category	Code
Captain, army	0110
Congressman	1111
Manager, accounting	1211
…	…

String similarity > t*

Output:

Code	Probability
3422	0.98

Phase 2

NN-based model

Output:
Top k predictions

Code	Probability
3422	0.85
2359	0.11
2634	0.032
…	…

t*… threshold for string similarity

Methodological approach

String Matching

Input String

"Golf-Coach"

Pre-processing:

"golf coach"

String similarity ∈ [0,1]
with string distance*

Category	Code	similarity
golfer	3421	0.4
coach, sports	3422	0.28
caddie, golf	9621	0.1
trainer, golf	3422	0.08
…	…	

similarity < t*

NN-based model

similarity ≥ t*

Output:
Code | Similarity

*string distance computed using R package `stringdist`

Text pre-processing

- All lower case
- Umlaut to non-umlaut (e.g. ü -> u)
- Remove stop words and special characters
- Consistent gender-specific word-endings (e.g. remove “-in”)

Methodological approach

Large Language Models

- R packages `keras` and `tensorflow`
  - Recurrent Neural Networks (**LSTM** and **GRU**)
  - **Transformer** Models

- String tokenization into **n-grams**

“golf coach” —3-grams→ `“gol” “f” “ “ “coa” “ch”` —Token ID vector→ `44,5,1,340,22,0,0,0,0,0`

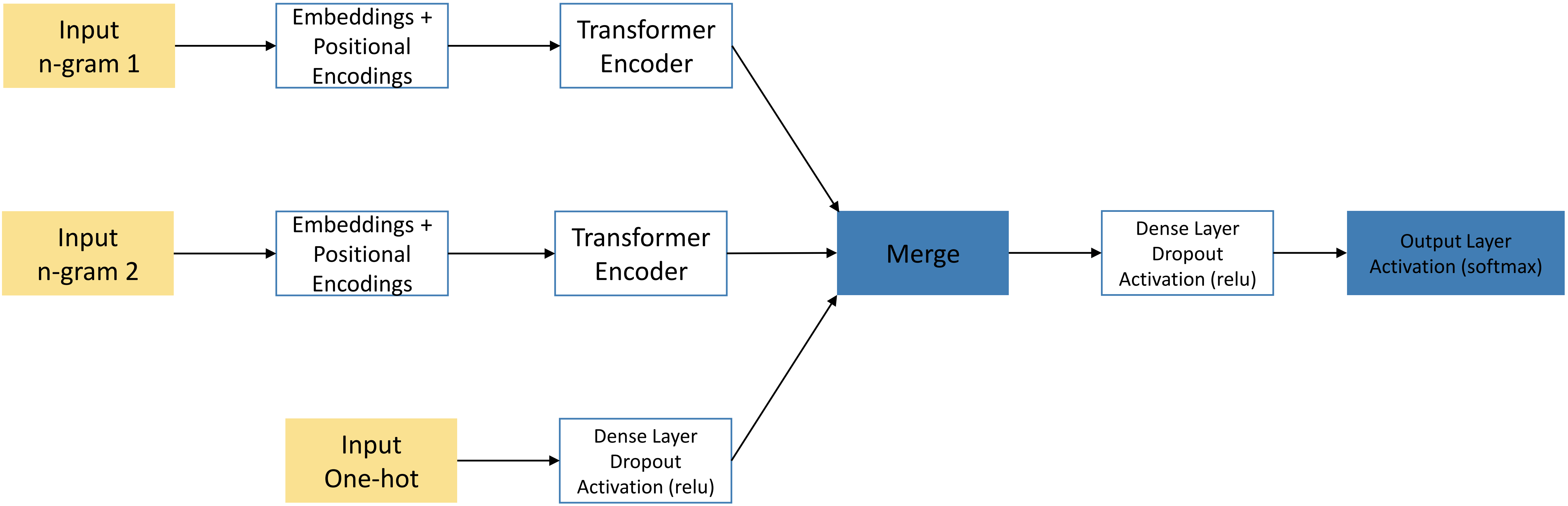
padding

fixed length

- **One hot encoding** for categorical variables and token IDs

Token0	Token1	Token2	Token3	Token4	Token5	Token6	…	Citizen_AT	Citizen_DE	…
1	1	0	0	0	1	0	…	1	0	…
1	1	0	1	1	0	0	…	0	1	…

Methodological approach
Example NN-Model Architecture



Other considerations

Pretrained Model

- First attempts with pre-trained LLMs from the huggingface
- Very large models (possible over-kill for the task)
- High computational cost -> no GPU available

Hugging Face

Train NN-based models from scratch

(turns out they outperform pre-trained LLMs)

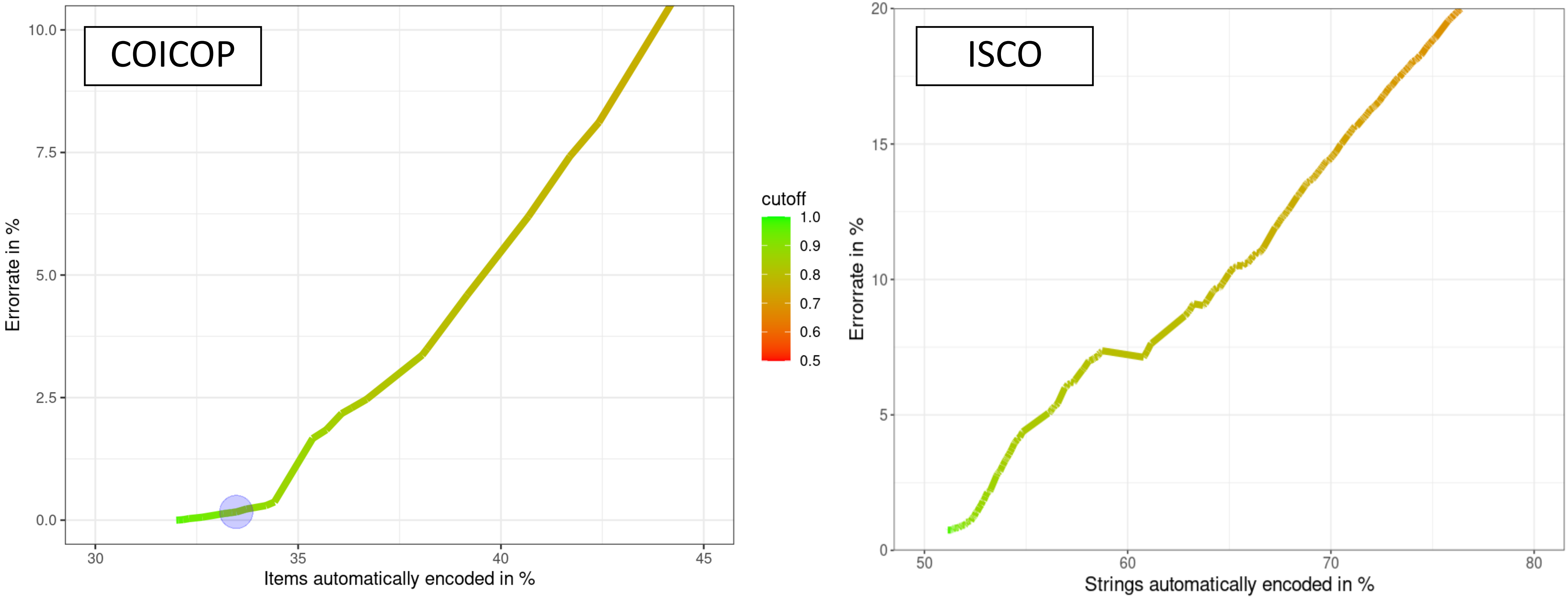
Other considerations
Hierarchical Modelling

- One Model per hierarchy level
  → Condition lower level models on predictions of higher models
- One model with multiple output (one per level)

→ Have not managed to achieve comparable results

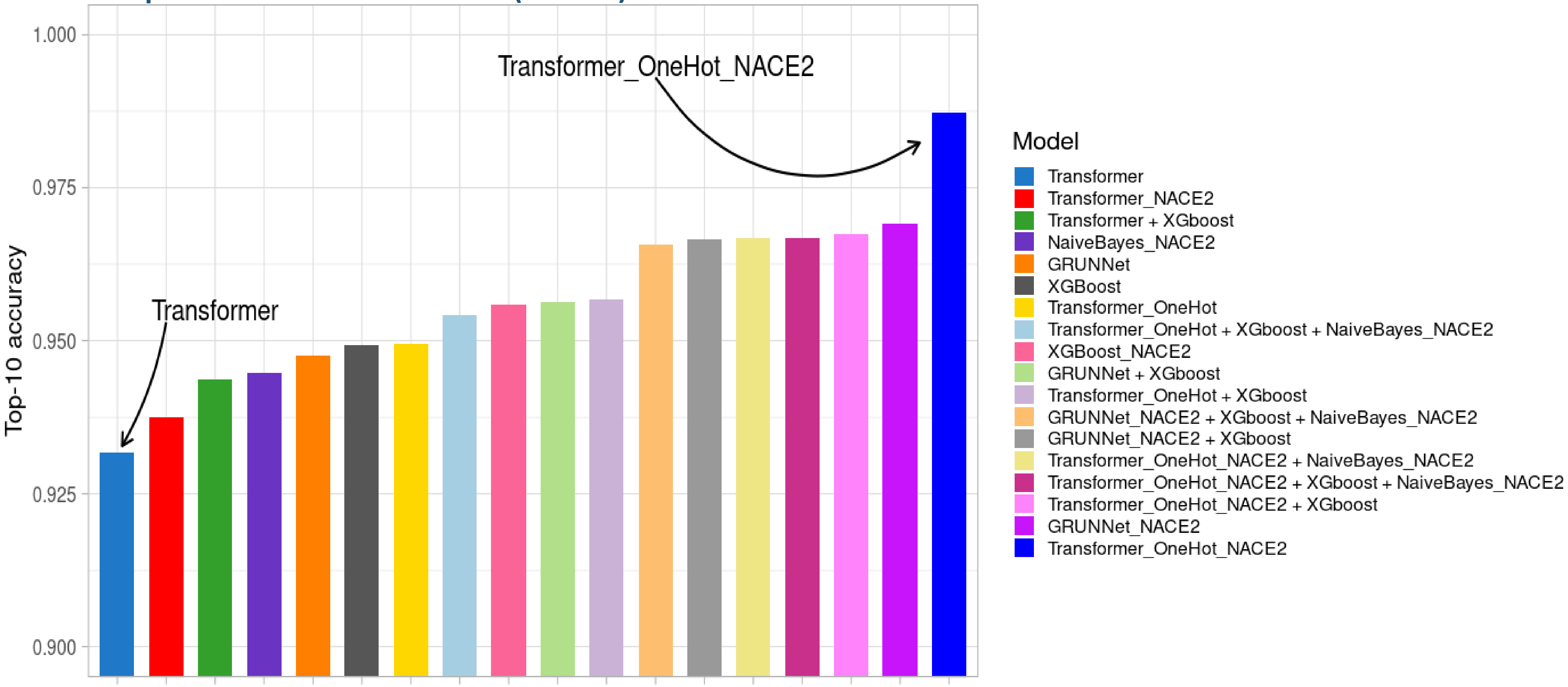
Results

Results
String Matching



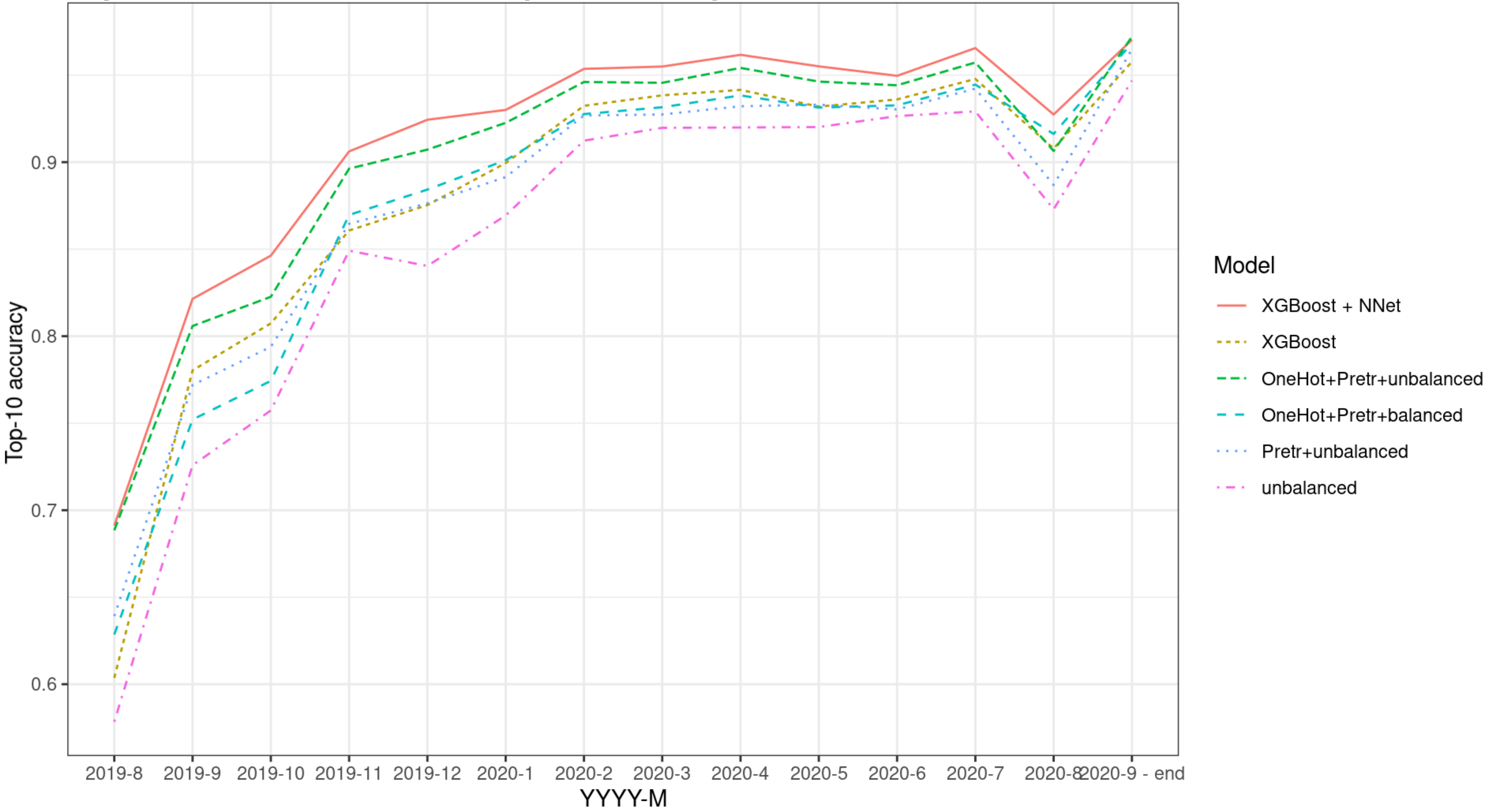
Model Results

Comparison of models (ISCO)



Model Results

Comparison of models (COICOP)



Results Overview

Code	% automatically encoded (error)	Top-5 accuracy	Top-10 accuracy
COICOP	33% (0.2%)	-	>90%
ISCO	38% (1%)	93%	96%
ISCED	15% (4%)	82%	87%
NACE	13% (2.7%)	56%	63%

ISCO Results per survey

Survey	Top-5 accuracy	Top-10 accuracy
MZ	76%	82%
JVS	81%	87%
SES	95%	97%

Deployment

Deployment
plumber API

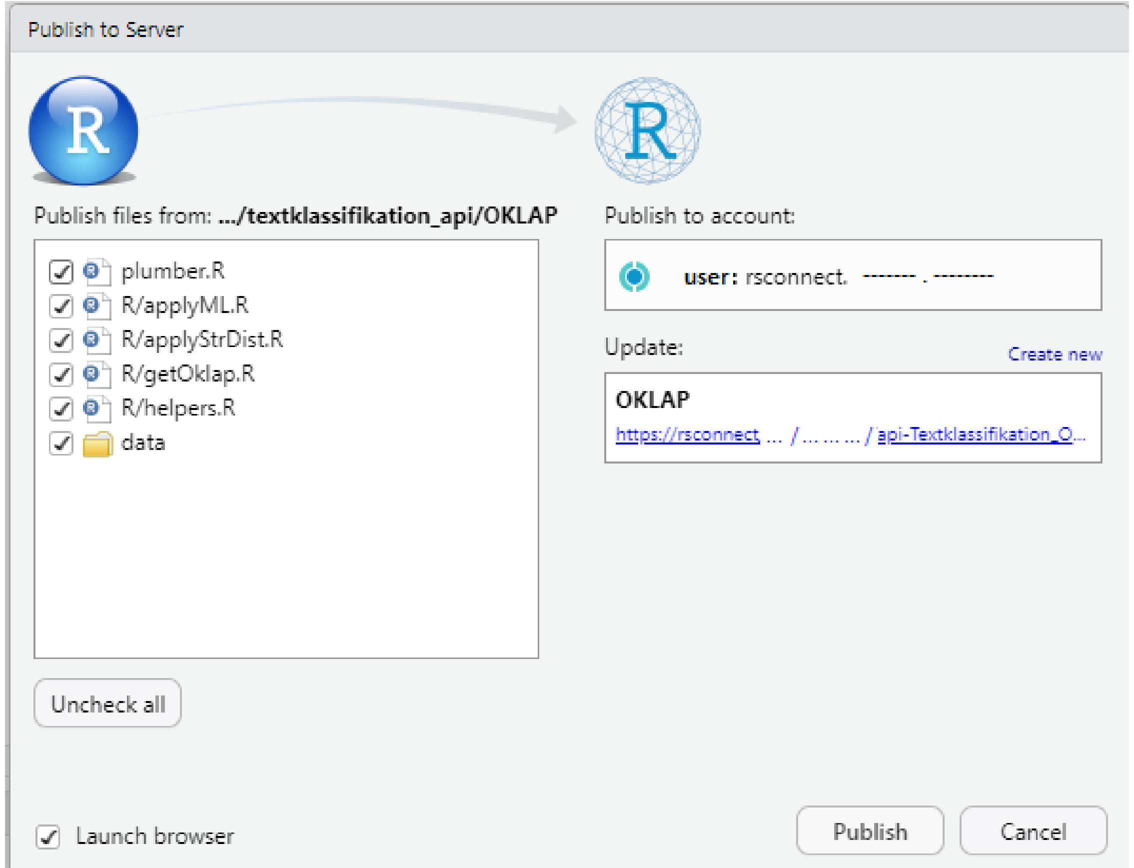
- **Send a request** with input data to the plumber API (json format)
- **Model** makes code **predictions** given input data
- API sends **results** back (json format)
- API is integrated in an App that lets users send requests to the API

- Deploy **one API for each standardized code** due to varying hyperparameters

Deployment

plumber API - publishing

- Hosting on **Posit Connect** integrated into RStudio IDE
- All employees with the API link have access (no access key necessary)
- Predictions are done in **batches**



Deployment

plumber API - requests

REQUEST

REQUEST BODY* application/json

Predictions for ISCO codes

EXAMPLE SCHEMA

```
{
  "top_n": 3,
  "input_text": [
    "golf coach"
  ],
  "Bildung": [
    "Bakkalaureat/Bachelor"
  ],
  "NACE2": [
    "85"
  ],
  "Anstellung": [
    "Missing"
  ],
```

API **request**

Response Status: OK:200
Took 208 milliseconds

RESPONSE RESPONSE HEADERS CURL

```
[
  {
    "text_input":
    "prediction": [
      3422,
      2359,
      2635
    ],
    "probabilities": [
      0.8576,
      0.0297,
      0.0112
    ]
  }
]
```

API **response**

Current Status

Code	Model	Status
COICOP	XGB+NNet	Production
ISCO	Transformer+XGB	Testing
NACE	Transformer	Testing
ISCED	Transformer+XGB	Testing

Conclusions and outlook

- LLMs used with top-k predictions work well for our classification use-cases
- Work in progress and potential to improve
- Expand work to other classification codes
- Include Hierarchical Structures into models
- KPIs for API monitoring

Thank you!