DaCy: A Unified Framework for Danish NLP

Kenneth Enevoldsen[1,2], Lasse Hansen[1,2,3] and Kristoffer L. Nielbo[1,2]

[1]*Interacting Minds Centre, Aarhus University, Denmark*
[2]*Center for Humanities Computing, Aarhus University, Denmark*
[3]*Department of Clinical Medicine, Aarhus University, Denmark*

Abstract

Danish natural language processing (NLP) has in recent years obtained considerable improvements with the addition of multiple new datasets and models. However, at present, there is no coherent framework for applying state-of-the-art models for Danish. We present DaCy: a unified framework for Danish NLP built on and integrated with SpaCy. DaCy uses efficient multitask models which obtain state-of-the-art performance on named entity recognition, part-of-speech tagging, and dependency parsing. DaCy contains tools for easy integration of existing models such as for polarity, emotion, or subjectivity detection. In addition, we conduct a series of tests for biases and robustness of Danish NLP pipelines through data augmentation. DaCy large compares favorably and is especially robust to long input lengths and spelling variations and errors. All models except DaCy large display significant biases related to ethnicity while only Polyglot shows a significant gender bias. We argue that for languages with limited benchmark sets, data augmentation can be particularly useful for obtaining more realistic and fine-grained performance estimates. We provide a series of augmenters as a first step towards a more thorough evaluation of language models for low and medium resource languages and encourage further development.

Keywords

Natural Language Processing, Low-resource NLP, Data Augmentation, Danish NLP

1. Introduction

Danish Natural Language Processing (NLP) has seen a recent rise in resources with the introduction of the Danish Gigaword Corpus [10], curated lists of Natural Language Processing (NLP) tools by DaNLP [4] and sprogteknologi.dk, and at least five pretrained neural language models [14, 22, 32]. Datasets and models are available for most common tasks such as Named Entity Recognition (NER), Part-of-Speech (POS) tagging, dependency parsing, sentiment analysis, and coreference resolution [4, 30]. However, no coherent, efficient and state-of-the-art framework exists for all fundamental NLP tasks. Models are developed and distributed as disjoint projects and often require diverging package versions and have idiosyncratic APIs. These factors complicate workflows and hamper further developments.

1.1. DaCy

With this motivation we present DaCy: an efficient end-to-end framework for Danish NLP with state-of-the-art performance on POS, NER and dependency parsing. DaCy fills the

---

CEUR Workshop Proceedings (CEUR-WS.org)

gap in Danish NLP by providing a consistent interface that is easily extendable and able to integrate other models. DaCy is built on SpaCy v.3 which comes with a range of advantages: the framework is optimized, user-friendly, and well-documented. DaCy includes three fine-tuned language models: DaCy small, based on a Danish Electra (14M parameters) [14]; DaCy medium, based on the Danish BERT (110M parameters) [22]; and DaCy large, based on the multilingual XLM-RoBERTa (550M parameters) [9]. All models have been fine-tuned to do POS tagging, NER, and dependency parsing in a single forward pass, which increases the efficiency of the model and allows for larger models at the same computational cost.

Besides models fine-tuned for DaCy, the package includes convenient wrappers to add other models to the pipeline. For instance, Danish models for detecting polarity, emotion, and subjectivity classification can be added in a single line of code, and any HuggingFace Transformers [34] model trained for sentence classification can be conveniently wrapped and included in the pipeline using utility functions. With this functionality, DaCy aims at being a unified framework for Danish NLP. All functionality is well-documented and covered by tutorials.[1]

1.2. Robustness & Evaluation

Fine-tuned language models are commonly evaluated by testing performance on a gold-standard benchmark dataset. The most commonly used benchmark for Danish is the DaNE dataset [16], which consists of the Danish Dependency Treebank [18], additionally tagged for NER. For languages with few benchmarks datasets, such as Danish, the performance stability and generalizability can not be reliably estimated [27]. For instance, the text included in DaNE was collected in the years 1983–1992 from both written and spoken domains [16]. Given the change of languages over time and the addition of new textual domains such as social media, this dataset is unlikely to be representative of the contemporary domains of application. For instance, models might not be sufficiently exposed to e.g. abbreviated names, spelling errors, or non-standard casing to correctly and robustly classify them. In this sense, the performance obtained on DaNE is unlikely to hold for real-world use cases.

To provide an additional layer of validation, we propose evaluating models on augmented gold-standard data. Data augmentation entails generating new data by slightly modifying existing data points [13]. Data augmentation techniques such as rotation and cropping are widely used in computer vision to reduce overfitting [29], and are becoming increasingly common in NLP [7]. The complex syntactic and semantic structure of text complicates the task of finding useful augmentations, but simple manipulations such as synonym replacement and random character swaps and deletions have been found to be particularly useful for supervised learning in low-resource settings [33].

Although data augmentation is most commonly used for increasing the amount of training data, it can just as well be used for evaluation purposes [27]. By augmenting a gold-standard dataset, we can evaluate model performance when exposed to data that more closely mimics real-life settings by adding spelling errors, more diverse names, or other manipulations. In section 2.2, we introduce a series of augmentations and evaluate the performance of Danish NLP pipelines on them.

The contributions of this paper are three-fold. 1) We introduce new state-of-art models for Danish dependency parsing, NER and POS. 2) We introduce the DaCy Python library as a unified framework for state-of-the-art NLP in Danish. 3) We evaluate Danish NLP pipelines

---

[1]See: https://centre-for-humanities-computing.github.io/DaCy/

Table 1

Performance of models finetuned for DaCy. Highest scores are in bold and second highest is underlined. WPS indicates words pr. second.

		POS	NER					Dependency Parsing		Speed
Framework	Model	Accuracy	PER	LOC	ORG	MISC	Avg. F1	UAS	LAS	WPS
DaCy large	XLM-RoBERTa	**98.39**	**95.53**	**83.90**	**77.82**	**80.16**	**85.20**	**90.59**	**88**	4311
DaCy medium	DaBERT	<u>97.93</u>	<u>89.62</u>	<u>83.09</u>	<u>67.35</u>	<u>70.69</u>	<u>78.47</u>	<u>87.88</u>	<u>85</u>	8335
DaCy small	Ælæctra Cased	97.69	87.36	81.95	63.83	70.68	76.55	86.45	83	**10671**
	DaELECTRA	97.40	82.80	77.39	63.01	66.95	73.16	85.20	82	9855
	DaConvBERT	97.23	85.08	78.26	61.76	66.93	73.77	84.61	81	<u>10029</u>

using data augmentation and provide directions for future model development.

2. Methods

2.1. Training

To train the candidate models for DaCy, all publicly available Transformer-based language models for Danish were fine-tuned on the DaNE corpus [16] using SpaCy 3.0.3 [15]. The models include 2 Danish ELECTRAs [8, 14, 32], the Danish ConvBERT [17, 32], the Danish BERT [11, 22], and the multilingual XLM-Roberta Large [9]. All models were trained with an input length of 10 sentences until convergence using similar hyperparameters on a Quadro RTX 8000 GPU. Adam was used as optimizer with hyperparameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$. Further, L2 normalization with $\alpha = 0.01$ and gradient clipping with $c = 1.0$ was employed. For increased efficiency, all models were trained with a multi-task objective [6, 28] on NER, POS, and dependency parsing. This allows the training of larger models at the same computational cost, but it is unlikely that multi-task training at this scale improves performance [25, 1].[2]

Table 1 shows the performance of all fine-tuned models evaluated on DaNE's test set. The three best performing models in each size category, XLM-Roberta, DaBERT, and Ælæctra Cased are included in DaCy as the large, medium and small, respectively. In line with previous findings [25, 5, 24], larger models tend to perform better with XLM-Roberta obtaining the best performance across the board.

2.2. Evaluation

To evaluate the robustness of DaCy and other Danish NLP pipelines, we assessed their performance on multiple augmented version of the DaNE test set. All Danish models are trained on the DaNE corpus which consists of a mix of textual data of both spoken and written origin from the years 1983–1992 [16], with the exception of Polyglot which is trained on entities extracted from Wikipedia [26]. As a consequence, the training data is rarely representative of the domain in which the models will be applied. For example, social media, contemporary news media, and historical texts have domain specific characteristics such as non-standard casing, a higher degree of typos, use of hashtags, and historic spelling such as upper-cased nouns [31, 3, 12]. While it is infeasible to test the models on all possible domains, some of these characteristics

---

[2]For a full list of models and training configurations see the config files on Github: https://github.com/centre-for-humanities-computing/DaCy/tree/main/training

can be modelled using data augmentation which can provide practitioners with an estimate of the potential shortcomings of the model. Further, data augmentation can be used to estimate biases against protected groups such as gender and ethnicity.

The augmenters presented here are not meant to be exhaustive, but rather a first step towards more thorough validation of new language models. We argue that the bar for inclusion of a new model should be set higher than a slight increase in benchmark performance. Language models are used in a variety of contexts which current benchmarks tasks, especially for low resource languages, do not capture. Our aim with these experiments is to provide an extra layer of insight into the performance of language models that more closely mimics naturalistic use cases, and encourage the development of further augmenters. Augmentation not only provides insights into when model performance breaks down, whether certain models are more suited for specific use-cases than others, but can also be used for identifying specific areas to improve upon.

The augmenters developed for this paper are designed in accordance with the SpaCy framework, and are thus not necessarily tied to DaCy or Danish in particular and can be used both during model validation and training. Comprehensive tutorials are provided on the DaCy Github repository.

We tested small, medium, and large SpaCy [15] and DaCy models, Stanza [23], Polyglot [26], NERDA [19], Flair, [3] [2], and DaNLP's BERT [4] on the DaNE test set augmented with the following augmenters:

1. Keystroke augmentation: substitute 2%, 5%, or 15% of characters with a neighbouring character on a Danish QWERTY keyboard.
2. ÆØÅ augmentation: substitute æ/Æ with ae/Ae, ø/Ø with oe/Oe, and å/Å with aa/Aa to simulate some historic text variations in Danish.
3. Lower-case augmentation: convert all text to lower-case.
4. Spacing augmentation: randomly remove 5% of all whitespace.
5. Name augmentations:
   a) Substitute all names (PER entities) with randomly sampled Danish names, respecting first and last names.
   b) Substitute all names with randomly sampled names of Muslim origin used in Denmark [21], respecting first and last names.
   c) Substitute all names with sampled Danish male names, respecting first and last names.
   d) Substitute all names with sampled Danish female names, respecting first and last names.
   e) Abbreviate all first names to the first character including a full stop.

The stochastic augmentations, i.e. name and keystroke augmentations, were repeated 20 times.

Previous evaluations of Danish NLP tools have used the gold-standard tokens instead of using a tokenization module. While this allows for easier comparison of the specific modules it inflates the performance metrics of the models and is unlikely to reflect the metric of interest, namely, the performance during application.[4] All models were tested using both their own tokenizer (if they have one) and the SpaCy tokenizer for Danish. The performance reported in

---

[3]As supplied by DaNLP.

[4]In our experiments, several of the Danish models performed worse using their own tokenizer.

Table 2

Performance of Danish NLP pipelines. Wall Time is the time taken by the model to go through the DaNE test set without augmentation. Stanza uses the spacy-stanza implementation. The speed of the DaNLP model is reported as provided by the framework, which does not utilize batch input. However, given the model size it can be expected to reach speeds comparable to DaCy medium. Empty cells indicates that the framework does not include the specific model.

Model	POS Accuracy	NER Person	Location	Organization	Misc	F1	F1 w/o Misc	Dependency Parsing LAS	UAS	Wall Time GPU/CPU
DaCy large	**98.37**	**93.33**	**84.88**	**76.49**	**80.16**	**84.39**	**85.65**	**88.44**	**90.85**	2.9 / 34.7
DaCy medium	<u>98.15</u>	89.86	83.96	64.47	70.09	77.67	79.68	<u>86.65</u>	<u>89.25</u>	1.8 / 9.9
DaCy small	97.75	87.98	79.23	60.58	64.82	74.18	76.98	84.03	87.63	1.9 / 2.6
DaNLP BERT		92.27	83.90	<u>71.13</u>		72.84	<u>83.20</u>			37.4 / -
Flair	97.80	<u>92.60</u>	<u>84.82</u>	61.29		70.49	81.09			2.0 / -
NERDA		92.35	81.52	65.96	<u>72.41</u>	<u>79.04</u>	80.85			2.5 / -
Polyglot	76.26	79.25	68.06	40.69		56.67	65.32			- / 3.8
SpaCy large	96.30	86.17	84.16	63.36	65.52	75.75	78.57	78.01	81.95	0.9 / 1.4
SpaCy medium	95.71	84.55	77.29	63.16	63.25	73.23	76.01	77.73	81.87	1.2 / 1.4
SpaCy small	94.80	78.92	69.04	53.49	61.54	67.11	68.61	74.03	78.68	1.4 / 1.5
Stanza	97.62							83.84	87.34	29.3 / -

section 3 uses the best peforming tokenization module for each pipeline. For all models except Stanza and Polyglot this was found to be the SpaCy tokenizer.

3. Results

Table 2 shows the overall performance of Danish NLP frameworks on POS, NER, and dependency parsing on the un-augmented DaNE test set. DaCy large obtains a new state-of-the-art on all three tasks, most notably on NER and dependency parsing. Regardless of model, performance for POS is stable around 98% accuracy. POS tagging has long been at this level, and obtaining greater accuracy has been argued to require updates to the training data rather than new architectures [20].

Tables 3 to 5 shows a detailed performance breakdown of the models on NER, POS, and dependency parsing on the augmented data described in section 2.2. Overall, spelling variations and abbreviated first names consistently reduce performance of all models on all tasks. Even simple replacements of æ, ø, and å lead to performance degradation. In general, larger models handle augmentations better than small models with DaCy large performing the best on all augmentations with the exception of lower-casing. DaCy medium, DaNLP's BERT, and NERDA are based on the uncased Danish BERT [22], and are consequently not affected by casing. The BiLSTM-based models (Stanza and Flair) perform competitively under augmentations and are only consistently outperformed by DaCy large.

On NER specifically, all models with the exception of DaCy large obtain significantly worse performance on Muslim names as compared to Danish names. The robustness of DaCy large likely stems from the multilingual pre-training and the model size. Similarly, DaCy small is robust to spelling errors and outperforms larger models such as DaNLP's BERT and NERDA, this is likely due to its well-curated training data [10]. DaNLP's BERT and NERDA models were found to severely under-perform if given longer input lengths. DaCy's models consistently perform slightly better with more context, but are not vulnerable to shorter input. Lastly, as expected, the lack of casing is especially detrimental for NER for the cased models, most

Table 3

NER performance of Danish NLP pipelines reported as average F1 scores excluding the MISC category on the test set of DaNE. Best scores are marked bold and second best are underlined. * denotes that the result is significantly different from the baseline using a significance threshold of 0.05 with Bonferroni correction for multiple comparisons. Danish names is considered the baseline for the augmentation of Muslim, female, and male names. Values in parentheses denote the standard deviation. NERDA limits input size to 128 wordpieces which leads to truncation on long input sizes and high rates of keystroke errors.

| | | Deterministic Augmentations | | | | |
| | | | | Input Length | | Names |
Model	Baseline	Æøå	Lowercase	5 sentences	10 sentences	Abbreviated
DaCy large	**85.6**	**83.5**	69.7	**86.5**	**86.5**	**80.1**
DaCy medium	79.7	73.1	79.5	80.9	80.4	76.3
DaCy small	77.0	74.7	48.0	77.0	78.2	69.4
DaNLP BERT	<u>83.2</u>	78.6	**83.1**	78.6	61.9	<u>78.1</u>
Flair	81.1	<u>80.2</u>	24.4	<u>81.0</u>	<u>80.9</u>	74.9
NERDA	80.9	74.8	<u>80.7</u>	73.7	53.8	76.4
Polyglot	65.3	61.4	55.3	64.8	64.2	40.2
SpaCy large	78.6	75.4	5.7	78.8	78.8	78.0
SpaCy medium	76.0	74.7	9.7	76.5	76.8	76.0
SpaCy small	68.6	66.9	4.8	68.0	68.0	63.8

| | Stochastic Augmentations | | | | | | |
| | Names | | | | Keystroke Errors | | |
Model	Danish	Muslim	Female	Male	2%	5%	15%
DaCy large	**86.2 (0.6)***	**86.0 (0.5)**	**86.2 (0.5)**	**86.2 (0.4)**	**82.0 (1.2)***	**76.9 (1.3)***	**61.3 (1.6)***
DaCy medium	80.3 (0.5)*	77.9 (0.8)*	80.3 (0.4)	80.2 (0.7)	65.5 (1.7)*	50.0 (1.6)*	25.8 (1.3)*
DaCy small	76.5 (0.9)	75.7 (0.7)*	76.7 (0.8)	76.6 (0.7)	70.7 (1.6)*	62.1 (1.5)*	41.3 (1.6)*
DaNLP BERT	82.9 (0.6)	81.0 (1.0)*	83.1 (0.5)	83.0 (0.7)	72.6 (1.2)*	60.9 (1.7)*	37.0 (1.5)*
Flair	<u>81.2 (0.7)</u>	79.8 (0.7)*	<u>81.4 (0.5)</u>	81.5 (0.5)	78.3 (0.9)*	73.5 (1.5)*	56.3 (1.7)*
NERDA	80.0 (1.1)*	78.1 (1.2)*	80.2 (0.8)	80.0 (0.8)	<u>70.7 (1.4)*</u>	<u>57.5 (1.4)*</u>	<u>31.1 (1.6)*</u>
Polyglot	63.1 (1.2)*	41.8 (0.7)*	61.2 (1.2)*	64.8 (1.2)*	57.4 (0.9)*	46.9 (1.9)*	24.7 (1.9)*
SpaCy large	79.5 (0.6)*	71.6 (1.1)*	79.8 (0.5)	79.4 (0.5)	72.1 (1.0)*	63.3 (1.5)*	44.9 (1.8)*
SpaCy medium	78.2 (0.7)*	69.2 (1.4)*	78.2 (0.7)	78.5 (0.8)	70.5 (1.3)*	64.2 (1.5)*	46.9 (1.6)*
SpaCy small	62.5 (1.6)*	57.8 (1.4)*	63.0 (1.1)	63.3 (0.9)	65.4 (0.7)*	60.5 (1.5)*	45.9 (1.6)*

notably Flair, the SpaCy models, DaCy large and DaCy small.

4. Discussion

This paper has introduced the DaCy models and presented a thorough evaluation of Danish NLP models on a battery of augmentations. DaCy models achieve state-of-the-art performance on Danish NER, POS, and dependency parsing, and are robust to augmentations such as keystroke errors, name changes, and lowercasing. The results from training DaCy underline three well-known trends in deep learning and NLP, 1) larger models tend to perform better, 2) higher quality pre-training data leads to better models, as illustrated by the superior performance of Ælæctra compared to DaELECTRA, and 3) multilingual models perform competitively with monolingual models [25, 35, 5].

Our experiments with multiple augmenters revealed different patterns of strengths and weaknesses across Danish NLP models. In general, larger models tend to be more robust to data

Table 4
POS performance of Danish NLP pipelines reported as accuracy on the test set of DaNE. Best scores are marked bold and second best are underlined. * denotes that the result is significantly different from baseline using a significance threshold of 0.05 with Bonferroni correction for multiple comparisons. Values in parentheses denote the standard deviation. NERDA limits input size to 128 wordpieces which leads to truncation on long input sizes and with a high degree of keystroke errors.

| | | Deterministic Augmentations | | | | Stochastic Augmentations | | |
| | | | | Input Length | | Keystroke Errors | | |
Model	Baseline	Æøå	Lowercase	5 sentences	10 sentences	2%	5%	15%
DaCy large	**98.4**	**97.5**	<u>95.5</u>	**98.5**	**98.4**	**95.5 (0.2)***	**91.1 (0.2)***	75.4 (0.6)*
DaCy medium	<u>98.2</u>	<u>96.5</u>	**98.1**	<u>97.8</u>	<u>97.9</u>	93.6 (0.3)*	86.5 (0.3)*	63.3 (0.6)*
DaCy small	97.7	95.4	95.4	97.6	97.7	93.1 (0.2)*	85.9 (0.4)*	62.5 (0.4)*
Flair	97.8	95.0	95.0	97.7	97.7	94.7 (0.2)*	89.8 (0.3)*	72.1 (0.4)*
Polyglot	76.3	71.6	75.6	75.7	75.6	71.7 (0.2)*	65.3 (0.3)*	49.4 (0.4)*
SpaCy large	96.3	92.4	91.5	96.3	96.3	91.5 (0.2)*	84.8 (0.4)*	66.2 (0.5)*
SpaCy medium	95.7	92.4	91.6	95.8	95.7	91.0 (0.3)*	84.5 (0.3)*	66.0 (0.5)*
SpaCy small	94.8	90.5	90.3	94.8	94.8	90.7 (0.2)*	85.3 (0.3)*	69.1 (0.4)*
Stanza	97.6	96.1	95.4	97.7	97.7	<u>94.8 (0.2)*</u>	<u>90.6 (0.3)*</u>	**75.6 (0.5)***

Table 5
Dependency parsing performance of Danish NLP pipelines on the test set of DaNE reported as LAS. Best scores are marked bold and second best are underlined. * denotes that the result is significantly different from baseline using a significance threshold of 0.05 with Bonferroni correction for multiple comparisons. Values in parentheses denote the standard deviation.

| | | Deterministic Augmentations | | | | Stochastic Augmentations | | |
| | | | | Input Length | | Keystroke Errors | | |
Model	Baseline	Æøå	Lowercase	5 sentences	10 sentences	2%	5%	15%
DaCy large	**88.4**	**86.2**	**87.0**	**88.3**	**88.3**	**83.7 (0.4)***	**76.6 (0.5)***	**53.6 (0.8)***
DaCy medium	<u>86.7</u>	<u>84.6</u>	<u>86.6</u>	<u>85.4</u>	<u>85.3</u>	79.9 (0.5)*	69.9 (0.7)*	41.1 (0.9)*
DaCy small	84.0	79.0	82.7	83.5	83.0	<u>76.8 (0.4)*</u>	66.2 (0.8)*	38.0 (0.6)*
SpaCy large	78.0	71.0	74.0	77.6	77.6	69.7 (0.5)*	59.3 (0.7)*	34.8 (0.7)*
SpaCy medium	77.7	71.2	73.8	77.4	77.4	69.6 (0.6)*	59.5 (0.6)*	35.3 (0.7)*
SpaCy small	74.0	65.9	70.4	74.1	74.1	67.5 (0.4)*	59.1 (0.5)*	38.2 (0.7)*
Stanza	83.8	80.2	82.5	83.9	83.9	79.0 (0.4)*	<u>71.9 (0.5)*</u>	<u>49.8 (0.9)*</u>

augmentations. Several models are highly sensitive to casing, which limits their usefulness on certain domains. Evaluating models on augmented data provides a more holistic and realistic estimate of the expected performance, and can reveal in which use cases one model might be more useful than another. For example, it might be better to use DaCy medium on social media as opposed to DaCy large as its performance is not affected by casing.

The purpose of the data augmentation experiments was to evaluate the robustness of Danish models and to open a discussion on how to present new models going forward. As more models are developed for low and medium resource languages, properly evaluating them becomes vital for securing robustness, transparency, and effectiveness despite limited benchmark sets. We do not posit data augmentation as the only solution, but demonstrate that it can effectively reveal performance differences on important factors such as casing, spelling errors, and biases related to protected groups. As researchers, we bear the responsibility for releasing adequately

tested and robust models into the world. With the increasing ease of deployment, users must be made aware of the level of performance they can realistically expect to achieve on their problem, and when to choose one model over another. Social media researchers should know that certain models are sensitive to casing, historians should know that some models handle old text variations such as ae, oe, aa poorly, and lawyers should be aware that models might not be able to identify abbreviated names as effectively. In this regard, transparency and openness as to when and how models fail are crucial measures to report. Such evaluation requires the development of infrastructure and tools, but is fast and easy to conduct once in place. For instance, it only takes 8 minutes to test DaCy large on all augmented datasets including bootstrapping. As part of the DaCy library, we provide several augmenters and utility functions for evaluation that integrate with SpaCy and encourage new NLP models to use and expand upon them. For the continued development of low and medium resource NLP in a direction that is beneficial for practitioners, it is vital to conduct more thorough evaluation of new models. We suggest these augmenters not as an evaluation standard, but as preliminary guiding principles for future development of NLP models for low and medium resource languages in particular.

Acknowledgments

References

[1] A. Aghajanyan, A. Gupta, A. Shrivastava, X. Chen, L. Zettlemoyer, and S. Gupta. "Muppet: Massive Multi-task Representations with Pre-Finetuning". In: *arXiv:2101.11038 [cs]* (2021). arXiv: 2101.11038. URL: http://arxiv.org/abs/2101.11038.

[2] A. Akbik, T. Bergmann, D. Blythe, K. Rasul, S. Schweter, and R. Vollgraf. "FLAIR: An Easy-to-Use Framework for State-of-the-Art NLP". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*. Minneapolis, Minnesota: Association for Computational Linguistics, 2019, pp. 54–59. DOI: 10.18653/v1/N19-4010. URL: https://aclanthology.org/N19-4010.

[3] T. Baldwin. "Social media: friend or foe of natural language processing?" In: *Proceedings of the 26th Pacific Asia Conference on Language, Information, and Computation*. 2012, pp. 58–59.

[4] A. Brogaard Pauli, M. Barrett, O. Lacroix, and R. Hvingelby. "DaNLP: An open-source toolkit for Danish Natural Language Processing". In: *Proceedings of the 23rd Nordic Conference on Computational Linguistics 2021)*. NoDaLiDa 2021. 2021.

[5] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. "Language Models are Few-Shot Learners". In: *arXiv:2005.14165 [cs]* (2020). arXiv: 2005.14165. URL: http://arxiv.org/abs/2005.14165.

[6]   R. Caruana. "Multitask learning". In: *Machine learning* 28.1 (1997), pp. 41–75.

[7]   J. Chen, D. Tam, C. Raffel, M. Bansal, and D. Yang. "An Empirical Survey of Data Augmentation for Limited Data Learning in NLP". In: *arXiv:2106.07499 [cs]* (2021). arXiv: 2106.07499. URL: http://arxiv.org/abs/2106.07499.

[8]   K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning. "ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators". In: *arXiv:2003.10555 [cs]* (2020). arXiv: 2003.10555. URL: http://arxiv.org/abs/2003.10555.

[9]   A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. "Unsupervised Cross-lingual Representation Learning at Scale". In: *arXiv:1911.02116 [cs]* (2020). arXiv: 1911.02116. URL: http://arxiv.org/abs/1911.02116.

[10]   L. Derczynski, M. R. Ciosici, R. Baglini, M. H. Christiansen, J. A. Dalsgaard, R. Fusaroli, P. J. Henrichsen, R. Hvingelby, A. Kirkedal, A. S. Kjeldsen, C. Ladefoged, F. Å. Nielsen, J. Madsen, M. L. Petersen, J. H. Rystrøm, and D. Varab. "The Danish Gigaword Corpus". In: *Proceedings of the 23rd Nordic Conference on Computational Linguistics*. 2021.

[11]   J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *arXiv:1810.04805 [cs]* (2019). arXiv: 1810.04805. URL: http://arxiv.org/abs/1810.04805.

[12]   A. Farzindar and D. Inkpen. "Natural language processing for social media". In: *Synthesis Lectures on Human Language Technologies* 8.2 (2015), pp. 1–166.

[13]   S. Y. Feng, V. Gangal, J. Wei, S. Chandar, S. Vosoughi, T. Mitamura, and E. Hovy. "A Survey of Data Augmentation Approaches for NLP". In: *arXiv:2105.03075 [cs]* (2021). arXiv: 2105.03075. URL: http://arxiv.org/abs/2105.03075.

[14]   M. Højmark-Bertelsen. "Ælæctra - A Step Towards More Efficient Danish Natural Language Processing". In: 2021. URL: https://github.com/MalteHB/-l-ctra/.

[15]   M. Honnibal, I. Montani, S. Van Landeghem, and A. Boyd. *spaCy: Industrial-strength Natural Language Processing in Python*. 2020. DOI: 10.5281/zenodo.1212303. URL: https://doi.org/10.5281/zenodo.1212303.

[16]   R. Hvingelby, A. B. Pauli, M. Barrett, C. Rosted, L. M. Lidegaard, and A. Søgaard. "DaNE: A named entity resource for danish". In: *Proceedings of the 12th Language Resources and Evaluation Conference*. Lrec '20. 2020, pp. 4597–4604.

[17]   Z. Jiang, W. Yu, D. Zhou, Y. Chen, J. Feng, and S. Yan. "ConvBERT: Improving BERT with Span-based Dynamic Convolution". In: *arXiv:2008.02496 [cs]* (2021). arXiv: 2008.02496. URL: http://arxiv.org/abs/2008.02496.

[18]   A. Johannsen, H. M. Alonso, and B. Plank. "Universal Dependencies for Danish". In: *Proceedings of the Fourteenth International Workshop on Treebanks and Linguistic Theories* (2015).

[19]   L. Kjeldgaard. "Nerda". In: GitHub, 2020. URL: https://github.com/ebanalyse/NERDA.

[20]   C. D. Manning. "Part-of-Speech Tagging from 97% to 100%: Is It Time for Some Linguistics?" In: *Computational Linguistics and Intelligent Text Processing*. Ed. by A. F. Gelbukh. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2011, pp. 171–189. DOI: 10.1007/978-3-642-19400-9_14.

[21] E. V. Meldgaard. *Muslimske fornavne i Danmark*. 2005. URL: https://nors.ku.dk/publikationer/webpublikationer/muslimske%5C%5Ffornavne/.

[22] J. D. Møllerhøj. *Danish BERT model: BotXO has trained the most advanced BERT model*. BotXO. 2019. URL: https://www.botxo.ai/blog/danish-bert-model/.

[23] P. Qi, Y. Zhang, Y. Zhang, J. Bolton, and C. D. Manning. "Stanza: A Python Natural Language Processing Toolkit for Many Human Languages". In: *arXiv:2003.07082 [cs]* (2020). arXiv: 2003.07082. URL: http://arxiv.org/abs/2003.07082.

[24] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. "Language models are unsupervised multitask learners". In: *OpenAI blog* 1.8 (2019), p. 9.

[25] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". In: *arXiv:1910.10683 [cs, stat]* (2020). arXiv: 1910.10683. URL: http://arxiv.org/abs/1910.10683.

[26] R. Al-Rfou', B. Perozzi, and S. Skiena. "Polyglot: Distributed Word Representations for Multilingual NLP". In: *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*. Sofia, Bulgaria: Association for Computational Linguistics, 2013, pp. 183–192. URL: https://aclanthology.org/W13-3520.

[27] M. T. Ribeiro, T. Wu, C. Guestrin, and S. Singh. "Beyond Accuracy: Behavioral Testing of NLP Models with CheckList". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, 2020, pp. 4902–4912. DOI: 10.18653/v1/2020.acl-main.442. URL: https://aclanthology.org/2020.acl-main.442.

[28] S. Ruder. "An Overview of Multi-Task Learning in Deep Neural Networks". In: *arXiv:1706.05098 [cs, stat]* (2017). arXiv: 1706.05098. URL: http://arxiv.org/abs/1706.05098.

[29] C. Shorten and T. M. Khoshgoftaar. "A survey on Image Data Augmentation for Deep Learning". In: *Journal of Big Data* 6.1 (2019), p. 60. DOI: 10.1186/s40537-019-0197-0. URL: https://doi.org/10.1186/s40537-019-0197-0.

[30] sprogteknologi.dk. *Sprogteknologi.dk*. 2021. URL: https://sprogteknologi.dk/.

[31] N. Tahmasebi. "A Study on Word2Vec on a Historical Swedish Newspaper Corpus". In: *Proceedings of the Digital Humanities in the Nordic Countries 3rd Conference, DHN 2018, Helsinki, Finland, March 7-9, 2018*. 2018, pp. 25–37. URL: http://ceur-ws.org/Vol-2084/paper2.pdf.

[32] P. T. Tamini-Sarnikowski. "Danish transformers". In: GitHub, 2020. URL: https://github.com/sarnikowski.

[33] J. Wei and K. Zou. "EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, 2019, pp. 6382–6388. DOI: 10.18653/v1/D19-1670. URL: https://www.aclweb.org/anthology/D19-1670.

[34] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush. "HuggingFace's Transformers: State-of-the-art Natural Language Processing". In: *arXiv:1910.03771 [cs]* (2020). arXiv: 1910.03771. URL: http://arxiv.org/abs/1910.03771.

[35] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel. "mT5: A massively multilingual pre-trained text-to-text transformer". In: *arXiv:2010.11934 [cs]* (2021). arXiv: 2010.11934. URL: http://arxiv.org/abs/2010.11934.