How to make the first move? Analyzing the impact of an acquisition model in deep active learning for a sequence labeling task

Ngoc Duyen Tanja Tu
Leibniz Institute for the German Language
Mannheim, Germany
tu@ids-mannheim.de

Abstract

Deep active learning is a promising method for training a tagger when resources are low. In this paper, we systematically analyse the impact of differently performing initial acquisition models on the performance of each successor model in a custom sequence labeling binary classification task. For this purpose, we compare the performance of 5 successor models. They are trained on data queried by a random sampler and two differently performing initial acquisition models. In this way, we get an impression of the impact of an initial acquisition model with a certain performance on the performance of the successor models.

1 Introduction

Deep active learning (DAL) is used in sequence labeling tasks like named entity recognition (NER; e.g. Kim, 2020) or part-of-speech-tagging (POS-tagging; e.g. Chaudhary et al., 2021). DAL is a combination of deep learning (DL; first model proposed by McCulloch and Pitts, 1943) and active learning (AL; first introduced by Lewis and Gale, 1994). In DAL a DL model is initially trained on a small amount of labeled data. This model is called an initial acquisition model (Tsvigun et al., 2022). The acquisition model is used to select data points (samples) to be annotated from an unlabeled data pool based on an algorithm, called a query strategy. The samples are selected by the acquisition model in a way that intends to increase the performance of the model. The selected samples are then manually annotated and added to the training data. A new model, called a successor model is then trained on the enlarged training data. The successor model is used as an acquisition model in the next DAL iteration. The iteration process is repeated until a predefined termination condition is reached, for example the model achieves a certain f-score.

We use DAL for the development of a monitor corpus that consists of questions on the German language, so-called language inquiries (Lang et al., 2023a) as in (1) *Heißt es* ***der Bayrische Ministerpräsident*** *oder* ***der bayrische Ministerpräsident****?* ['Is it the **bavarian Minister-President** or the **Bavarian Minister-President**?'][1]. Language inquiries serve as a primary source of authentic language data for a variety of research questions (Breindl, 2016). The core of the corpus is a collection of approx. 50,000 language inquiries sent by email from (supposed) laypeople to a language consulting service between 1999 and 2019. To ensure that researchers have access to the data points from the corpus that are relevant to their research question, we tag the corpus in different ways (Lang et al., 2023b). Annotating a subset of 500 randomly extracted language inquiries from our corpus showed that 98% of the data contain examples. In most cases, the examples have the pattern as the bold print in (1) (see Section 3.1). For this reason, in this paper, we focus on this category of examples, that we call "list of alternatives". We train a tagger that detects the span of a language inquiry that is a list of alternatives.

The annotations obtained provide an interesting insight for linguists: Extracting the most frequent list of alternatives in language inquiries can help to identify standardization gaps in grammar. Moreover, the list of alternatives can be used in grammatical online resources to make it easier for users to find information about grammatical phenomena: Rather than having to be familiar with the terminology to look up grammatical information, users could click on list of alternatives that matched their question.

As we work with copyright-protected data, we cannot rely on crowd-sourcing to annotate the data. However, the spans to be tagged have a specific pattern: The alternatives listed in the language

[1]Note that some translations are not in correct English but given in a way that the semantic meaning of the question becomes clear.

inquiries are very similar to each other (see (1)). Since the samples to be annotated show a specific pattern that is visible on the surface structure, it can be assumed that a model does not need a large amount of training data to learn this pattern. For this reason, DAL seems to be a suitable method. Although the spans to be annotated seem easy to tag because they all show a specific pattern, in most cases it is not possible to detect them with a rule-based approach (see Section 4.1). One reason is that the occurrences of the spans are not limited to certain positions in the language inquiries. Additionaly, the length of the spans to be tagged can vary greatly: An annotation can span characters, words, multiword phrases or whole sentences. We therefore use DAL for our task for two key reasons: firstly, a rule-based approach is of limited use and secondly, DAL does not require a huge amount of labeled data to train a model.

The main contribution of this paper is a systematic analysis of the impact of the acquisition model on the performance of each successor model. We identified for our task that the performance of the successor model depends on the initial acquisition model. The insights of this study could be transferred to similar sequence labeling tasks like automatic tagging of paronym candidates. Moreover, unlike many other studies on DAL (Zhang et al., 2022), this task is not a simulation, so it is possible for us to address aspects like real annotation cost and time.

The paper is structured as follows: In Section 2 we give an overview of work relevant to our task. After that, in Section 3 we explain our experimental setup and present our results in Section 4. Finally in Section 5 we then summarise our findings and identify open questions.

2 Related work

As we are using DAL for a sequence labeling task, we only focus on papers that take a similar approach, in order to apply the results to our work. Studies with a textual data basis have shown that DAL models perform just as well as DL models, although significantly less data is used for training: Mirbostani et al. (2023) trained a DAL model for morphological inflection and morphophonological processing in Cairene Egyption Arabic. Although the model is only trained on 30% of the data in comparison to the state-of-the-art model it performs just as well. Kim (2020) demonstrated for slot filling that a DAL model trained only on 15% of the data compared to the best-performing DL model still achieved 98-99% performance. These results indicate that DAL sounds promising for sequence labeling tasks.

Studies on the ideal implementation of DAL focus mainly on the selection of the best query strategy. Query strategies can be divided into uncertainty-based algorithms, diversity-based algorithms and hybrid algorithms. Uncertainty-based algorithms use the acquisition model to predict the labels for the data points of the unlabeled data pool. The data points are then ranked by the confidence score of the model and the ones with the lowest confidence scores are selected for manual annotation. Diversity-based algorithms sort the data points of the unlabeled data pool by their similarity according to different strategies. The most different data points are then selected for manual annotation. Hybrid algorithms are a combination of an uncertainty-based and a diversity-based algorithm.

Chaudhary et al. (2021) developed a suitable query strategy for a DAL model trained for POS-tagging. However, the proposed strategy cannot be applied to our task because it considers the confusion caused by different annotations for the same token. For example, the German word *die* ('the') could either be a pronoun or an article. In our task the probability is the same that a given token is part of a list of alternatives or not. Mirbostani et al. (2023) showed that using a suitable query strategy, in their case: an uncertainty-based algorithm, outperforms the model which is trained on randomly selected samples from the unlabeled data pool. They could not demonstrate that combining uncertainty and diversity query strategies achieve better results. In contrast, Kim (2020) showed that a combination of uncertainty and diversity query strategies yield better results in DAL for NER and slot filling than pure uncertainty or pure diversity query strategies as well as random sampling.

Radmard et al. (2021) focused on reducing the cost of annotating the selected samples from the unlabeled data pool by only querying subsequences of each unlabeled sentence for NER. Annotated labels are then propagated to the unlabeled data pool if the subsequence is the same. Using subsequences in our task is not feasible because in our case the probability is the same that a given subsequence is part of an annotation or not, unlike in NER.

While the works listed above have trained DL

models from scratch, Shelmanov et al. (2021) used deep pre-trained models and showed that the latter outperforms the former.

None of these works analysed the impact of the initial acquisition model on the performance of the successor models. This shortcoming in the research to date will be addressed in this paper.

3 Experimental setup

3.1 Data preparation

The manually labeled data set consists of 500 language inquiries. They have been randomly extracted from our corpus of approx. 50,000 language inquiries. The vast majority of questions contain explicit examples (see for example: (1)-(14)). The data set was annotated in three steps:

(i) First, two linguists annotated the examples found in the data with the following four categories:

- "list of alternatives", e.g. (1),
- "sentence", e.g. (2) *Wie schreibe ich diesen Satz richtig: **Er ist am Witze erzählen**".* [...] ['How do I write this sentence correctly: **"He is telling jokes**".'],
- "expression", e.g. (3) *Eigentlich sind Wörter wie "kein" und "einzig" nicht stiegerbar* [sic]. *Allerdings frage ich mich, wie dann der Ausdruck **"in keinster Weise"** zu erklären ist.* [...] ['Actually, words like "none" and "only" are not gradable. However, I wonder how the expression **"in *nonest manner"** can be explained.'] and
- "other" if an example does not fit in one of the categories.

The classification is done in order to develop a strategy based on the examples per category with which the examples can be tagged automatically. The distribution of the categories is as follows: About 50% of the examples can be assigned to the category "list of alternatives", about 23% to "sentence", about 18% to "expression", about 6% fall in the category "other" and about 2% of the language inquiries do not contain an example at all. As "list of alternatives" is the most common category, we have opted for this paper to automatically tag it.

(ii) The language inquiries were tokenized by splitting at white space. Each token of a language inquiry tagged with "list of alternatives" in (i) were annotated by the two annotators. Every token was labeled according to the BIO format with "B-part-of-a-list" for the beginning of a span, "I-part-of-a-list" if the token is inside a span or "O" if the token is not part of a span. The tokens of language inquiries tagged with other categories than "list of alternatives" were tagged with "O". 90% of the annotated list of alternatives consists of two alternatives, the remaining 10% of 3 to 6.

Set	Token total	Token alternative
Train	12,799	1,248
Val	2,180	168
Test	2,753	345

Table 1: The distribution of all tokens and tokens that are part of a list of alternatives in the sets.

The inter-annotator-agreement is very high with a Fleiss' Kappa of 0.99. The two annotators discussed differing annotations and determined the final annotation.

500 language inquiries comprising 17,732 tokens were tagged and randomly split in training (approx. 70%), validation (approx. 15%) and test set (approx. 15%) in a way, so that no question is split in more than one data set (see table 1). Quotation marks are removed from the questions to prevent overfitting because 18% of the lists of alternatives are written in between quotation marks. Additionally, 36% of the language inquiries containing a list of alternatives either do not have the alternatives in quotation marks but other tokens or closing quotation marks are missing. In 46% of the data quotation marks are not used at all.

The remaining approx. 49,500 language inquiries from the corpus form the unlabeled data pool and will be shown to the model during each DAL iteration. 40 language inquiries are selected in each iteration increasing the manually labeled data set by approx. 10% per iteration. The unlabeled data pool was split into four due to the RAM size. Thus, per iteration four unlabeled data pools were shown in succession to the acquisition model and 10 language inquiries were selected from each. Due to low annotation power, only one linguist annotated the selected language inquiries per iteration.

3.2 Task

Our model was trained to tag spans of list of alternatives proposed in language inquiries. The alternatives listed can be:

- different spellings, e.g. (4) ***...sie spiegeln den besonderen Charme des Schlosses wider.*** *Oder?* ***...sie spiegeln den besonderen Charme des Schlosses wieder.***
  ['**...they mirror the special charm of the castle.** Or? **...they *miror the special charm of the castle** '],

- syntactic alternatives, e.g. (5) [...] ***heiße Himbeeren mit Vanilleeis, heiße Himbeeren auf Vanilleeis.*** [...]
  ['**Hot raspberries with vanilla ice cream, hot raspberries on vanilla ice cream**. [...]'],

- (potential) synonyms, e.g. (6) *gibt es das Wort* ***konzeptionalisieren/Konzeptionalisierung*** *als Alternative zu* ***konzeptualisieren/Konzeptualisierung.*** [...]
  ['Does the word ***conceptionalise/*conceptionalisation** as an alternative to **conceptualise/conceptualisation** exist. [...]'],

- a list of several possibilities, e.g. (7) [...] *Heißen derartige Läden nun* ***Asiashop, Asienladen*** *oder gar* ***Asialaden****?*
  ['[...] Are such stores now called **Asia shop**, **Asian store** or even **Asia store**?'].

The alternatives can be presented in detail as in (4) or shortened as in (8) [...] *Ich arbeite in einer Firma, deren* ***spannende(n)*** [...] *Aufgaben sehr reizvoll sind.* ['I work for a company whose ***excit(e)ing** [...] tasks are very appealing.']. Note that in (8) the alternative is given within the parenthesis. In some cases, the alternatives are connected by *oder* ('or') as in (1) and (4) but in other cases, there is no connector at all.

The task is modeled as a sequence labeling, binary classification task. Every token will be labeled according to the BIO format (see Section 3.1). The length of spans can be whole sentences as in (4), words as in (6) or characters as in (9) *im Duden steht Schmant mit* ***t****, auf den Sahnetöpfchen* [...] *mit* ***d****.* [...]['in the dictionary Schmant is written with a **t**, on the cream pots [...] with a **d**.']

The test set (see Section 3.1) contains 57 lists of alternatives, including 30 language inquiries on spellings, 11 on syntactic alternatives, 13 on synonyms and 3 lists of several possibilities.

3.3 Model

We use Flair sequence tagger (Akbik et al., 2019) as according to their GitHub page[2] many trained Flair sequence tagger models for different NLP tasks are state-of-the-art.[3]

The sequence tagger consists of the following layers: an embedding layer, an LSTM layer [4] and a CRF-layer.

We have left the default values of the hyperparameters (see table 2) except for the embeddings: We test two different pre-trained embedding settings for the initial acquisition model:

- The pre-trained German BERT model[5] dbmdz/bert-base-german-cased[6] which achieves better performance than google-bert/bert-base-german-cased on NER[7] as well as in our study.
- German non-contextualized FastText embeddings[8] stacked on German contextualized Flair forward and Flair backward embeddings[9] as suggested in Akbik et al. (2019).

BERT (Devlin et al., 2019) and Flair embeddings (Akbik et al., 2018) are trained differently: while BERT learns structures on token-level, Flair embeddings learn structures on character-level. On the one hand character-level models are better in handling typing errors (Gao et al., 2021), which is a big advantage for our non-normalized data. On the other hand, the span to be tagged is mostly a sequence of tokens. For these reasons, either BERT or Flair embeddings could be beneficial for our task.

[2] https://github.com/flairNLP/flair

[3] We also finetuned the models https://huggingface.co/google-bert/bert-base-german-cased, https://huggingface.co/dbmdz/bert-base-german-cased and https://huggingface.co/distilbert/distilbert-base-german-cased for our task to use as initial acquisition models but the f-scores are quite low with 0.29, 0.36 and 0.08.

[4] We also tested the performance of the models with a BiLSTM layer but the f-scores are lower than 0.15.

[5] We are aware that BERT only processes 512 subword tokens. Since only about 0.28% of our data basis is longer than this threshold, we accept the loss.

[6] https://huggingface.co/dbmdz/bert-base-german-cased

[7] https://github.com/stefan-it/fine-tuned-berts-seq

[8] https://flairnlp.github.io/docs/tutorial-embeddings/classic-word-embeddings

[9] https://flairnlp.github.io/docs/tutorial-embeddings/flair-embeddings

Hyperparameter	Value
Hidden size	256
Number of LSTM layers	1
Dropout	0.0
Classifier	Softmax + CRF
Mini batch size	32
Activation function	tanh
Max epoch	100
Initial learning rate	0.1
Patience	3
Annealing factor	0.5
Optimizer	SGD
Learning rate	Learning rate decay

Table 2: The default values of the hyperparameters.

Many aspects during training are set randomly, e.g. weight initialization. This can affect the performance of a model significantly (Reimers and Gurevych, 2017). For this reason, we trained the initial acquisition model five times.[10]

3.4 Active learning

We use the SeqAL framework[11] to perform active learning with the Flair sequence tagger. Various considerations were taken into account when selecting the query strategy: Choosing a pure uncertainty-based query strategy could result in samples that represent rare edge cases, that would make it difficult for the model to generalize from them. In addition to that, the model could select almost only similar samples that would result in redundant cases being annotated. Hence, a hybrid method could result in better performance from the model (Ren et al., 2022). Based on these arguments, we chose a hybrid query strategy.[12] As uncertainty-based method we chose maximum normalized log-probability (MNLP) which performs better than least confidence because it has no bias towards choosing longer sentences (Shen et al., 2017). The confidence score for each prediction, calculated by Viterbi loss, is used for this strategy. As diversity sampling (DS) method, we chose distribute similarity as it operates on token-level to compare the similarity of samples with each other, which is fitting for our task. For this method, the tokens of a sample are first embedded and the similarity is than calculated based on the cosine similarity. In contrast, cluster-based strategies are not suitable because we only tag one class. For comparison, we also selected samples randomly.

4 Results

4.1 Initial acquisition model

We report the performance of each model as f-scores only for the positive spans so the imbalance between the positive and negative class does not distort the results. We implemented a simple rule-based tagger as a baseline which detects the most frequent kind of list of alternatives: spelling variants (see Section 3.2). The rule-based tagger is only able to detect three types of spelling variants listed in language inquiries containing the token *oder* ('or'):

(i) the tagger checks if the token preceding *oder* and the token following *oder* in the language inquiry have a Jaro-Winkler similarity greater or equal 0.80.[13] If this is the case the tokens are tagged as "part-of-a-list". This way, cases like (10) ***Essensmarke*** *oder* ***Essenmarke***. *Und warum?* ['**Foodstamp** or **foodsstamps**. And why?'] are tagged.

(ii) The tagger checks if Token A preceding *oder* is a substring of Token B following *oder* and if the length of Token A is shorter than Token B. If this is the case, Token C preceding Token A is concatenated with Token A. If the concatenated token equals Token B Token A, B and C are tagged as "part-of-a-list". This way, cases like (11) ***hervor lugen*** *oder* ***hervorlugen*** ['**(to) peek out** or **(to) peekout**'] are tagged.

(iii) Analogous to (ii) except for the following token, to tag cases like (12) ***hinunterfallen*** *oder* ***hinunter fallen*** ['**(to) falldown** or **(to) fall down**'].

In addition, a random baseline was implemented, which randomly assigns "part-of-a-list" to 13% of the tokens and "not-part-of-a-list" to 87% as the test data show this distribution.

Five random seeds were set for the training of the sequence tagger with two different embedding configurations as described in section 3.3. The average performance of the models are reported in table 3.

[10] We only trained the initial acquisition model fives times and not every successor model as we wanted to keep the initial random weights constant for each acquisition model per iteration. Whether this actually has an effect on the performance of the successor models has to be clarified.

[11] https://github.com/tech-sketch/SeqAL

[12] We also experimented with selecting the samples only with MNLP and only with DS but these models performed worse than with a combination of MNLP and DS.

[13] We tested different thresholds and learned that 0.80 works best.

Embedding	Prec	Rec	F-score
flair_emb	0.39	0.05	0.09
BERT	0.51	0.38	0.44
rule_based	0.91	0.06	0.11
random_baseline	0.15	0.15	0.15

Table 3: The performance of a rule-based baseline, of a random baseline and of sequence taggers with different embeddings on average of five random seeds.

Model	Prec	Rec	F-score
BERT_0	0.46	0.35	0.40
BERT_1	0.46	0.38	0.42
BERT_2	0.56	0.36	0.44
BERT_3	0.51	0.41	0.45
BERT_4	0.55	0.41	0.47

Table 4: The performance of each initial acquisition model trained with BERT and five different random seeds.

The sequence tagger trained with Flair embeddings performed worst, even worse than the random baseline and the simple rule-based baseline. It can be deduced that character-level embeddings are not suitable for our task, so we must use token-level embeddings. The sequence tagger trained with BERT performed by far the best. For this reason, we decided to use the sequence tagger trained with BERT and different random seeds for the analysis of finding the ideal acquisition model. As shown in table 4 the f-score of the models trained with different random seeds range from 0.40 to 0.47. We decided to use BERT_0, the worst performing initial acquisition model, and BERT_4, the best performing initial acquisition model, in our DAL approach. The hypothesis is that the worst initial acquisition model is not as confident as the best and therefore selects more helpful samples.

4.2 Performance of the successor model

The overlap of selected samples of each acquisition model initialized with BERT_0 and BERT_4 averages only 33% per iteration. There is no overlap of selected samples between each acquisition model and the random sampler. The differently performing acquisition models therefore actually select different samples from the unlabeled data pool per iteration. Thus, in a next step it was possible to analyse if one of the two differently performing acquisition models selected better samples leading to a better successor model.

This was indeed the case: The performance of the successor models per iteration (see figure 1 and 2) are different depending on the model that selected the training data. Figure 1 shows the f-score for each successor model initialized with BERT_0 with different training data selected by its subsequent acquisition models, the acquisition models initialized with BERT_4 (i.e. the best initial acquisition model) and a random sampler. Figure 1 shows that the model achieves the best f-score of 0.65 (precision: 0.67, recall: 0.64) in iteration 19 when trained on the samples selected by its subsequent acquisition models. This specific successor model from iteration 19 is referred to BERT_0_best in the following. In contrast, the model only achieves 0.61 as the highest f-score in iteration 20 when trained on the samples selected by the acquisition models initialized with BERT_4. Figure 2 seems to confirm our hypothesis from Section 4.1: trained on data selected by the acquisition models initialized with BERT_0, i.e. the worst initial model, it achieves the overall highest f-score of 0.67 (precision: 0.70, recall: 0.64) in iteration 20. This specific successor model is referred to BERT_4_best in the following. However, the difference in performance is not as pronounced: trained on the samples selected by its subsequent acquisition models, the model achieves an f-score of 0.66 (precision: 0.66, recall: 0.66) in iteration 20. Still, it performed worse.

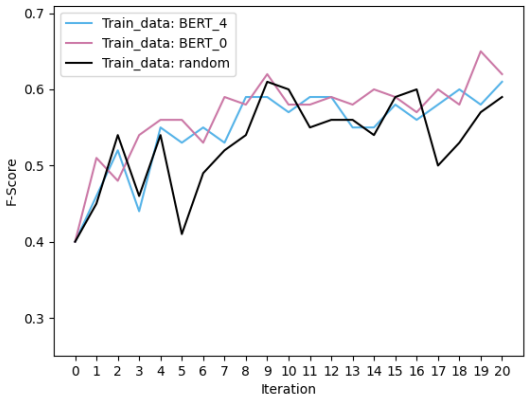

Figure 1: The f-score for all successor models initialized with BERT_0 and trained on data sampled by three different models per iteration.

To summarize: The best successor model of BERT_0 is achieved when the model is trained on data selected by its subsequent acquisition models in iteration 19 (BERT_0_best). The best suc-

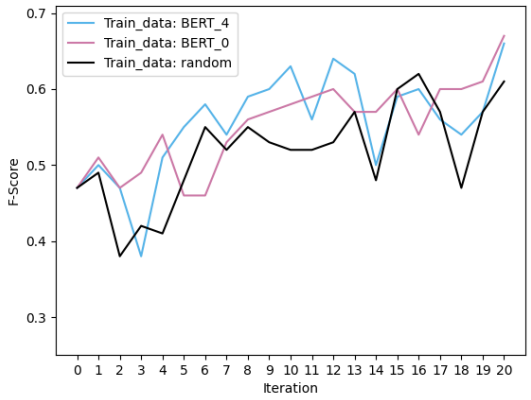

Figure 2: The f-score for all successor models initialized with BERT_4 and trained on data sampled by three different models per iteration.

cessor model of BERT_4 is achieved when the model is also trained on data selected by the acquisition models initialized with BERT_0 in iteration 20 (BERT_4_best). An error analysis of these two models shows that both models tend to overfit. Both models tend to tag spans that match a pattern similar to a list of alternatives, for example:
(13) *Zusammenstellung einer* ***14-tägigen all inclusive Gruppenreise****. Wie schreibt man* ***all inclusive Gruppenreise*** *nach den neuen Regeln,* [...]
['Compilation of a **14-day all inclusive group tour**. How to write **all inclusive group tour** according to the new rules, [...]'].
Conjunctions are in most cases not part of a list of alternatives, that is the reason why BERT_0_best overfits and tends to not tag a conjunction even if it is part of an alternative, e.g. *oder* ('or') in the first sentence in (14) ***..., danach das Fleisch medium oder durchbraten.*** *Oder:* ***..., danach das Fleisch medium- oder durchbraten.***
['**..., then roast the meat medium or well done.** Or: **..., then roast the meat medium- or well done.**']. This is reflected in the slightly lower precision value of this model compared to the other. In addition to that, BERT_0_best tags tokens that are not alternatives more often than BERT_4_best. However, we cannot make out a rule why the tokens are falsely tagged by BERT_0_best.

The f-scores seem low, but if we look at the false positives like in (13) we can see that in many cases examples in the language inquiries were tagged that are not list of alternatives but fall into other categories (see section 3.1): For BERT_0_best it is 109 out of 127 false positives and for BERT_4_best it is 95 out of 109 false positives. These spans are of great interest for us, even if they do not correspond to the task at hand because we want to extract all examples from the language inquiries for the corpus development.

Model	0_train	4_train	random
BERT_0	0.65	0.61	0.61
BERT_1	0.64	0.61	0.60
BERT_2	0.63	0.65	0.63
BERT_3	0.70	0.62	0.62
BERT_4	0.67	0.66	0.62

Table 5: The best f-scores through all iterations for each random seed and each training data set sampled by the acquisition models initialized with BERT_0 (0_train), BERT_4 (4_train) and a random sampler (random).

4.3 Train other models on the sampled training data

We see that the best performance results from training the best initial model with the samples selected by the acquisition models initialized with the worst initial model. The fact that the training samples of the worst initial model lead to a better performance is not limited to the two models evaluated in detail, BERT_0_best and BERT_4_best. Table 5 shows the best f-scores of the successor models of all five initial models differentiated according to the two training sets. Except for the best performing successor model initialized with BERT_2 the models achieve a better performance with the training data sampled by the acquisition models initialized with BERT_0, i.e. the worst initial model. It can also be seen, that the best f-score achieved by the models trained on the data sampled by the acquisition model initialized with BERT_4, i.e. the best initial model, is in most cases as high or similarly high as the models trained on the data randomly sampled. Although the models trained on the data sampled by the acquisition models initialized with BERT_0 achieve mostly the highest f-score, the models require an average of 18 iterations to achieve it. The models trained on the data sampled by the acquisition models initialized with BERT_4 achieve their highest score on average after 16 iterations and the models trained on the data sampled by the random sampler need 13 iterations to achieve their highest score. It is therefore a question of time resources as to whether performance can be lost.

4.4 Time investment

The annotation of the initial data set (all three steps) took about 14 hours in total for both annotators. Overall annotating the selected samples for each iteration in three DAL approaches took about 13 hours. Note that only one person annotated the selected samples. The training of each model took about 6.5 hours (approximately 20 minutes per iteration for each model) with 1 NVDIA Tesla V100 GPU. We set a cut at iteration 20 because by then we reached the time we planned to invest in AL.

5 Conclusion

In our contribution, we analysed the impact of the acquisition model to the performance of each successor model. We have found that the performance of the successor models in our task differ depending on which model has selected the data to be trained with. Based on this finding, we suggest the following approach:

(i) train several initial models with different random seeds,

(ii) select the worst performing one as acquisition model,

(iii) train the other initial models from (i) on the data selected by the subsequent acquisition models defined in (ii) and

(iv) use the best performing model out of all trained successor models from any iteration to solve your task.

However, our analysis is restricted to only one specific task using German language data and five random seeds. For this reason, the experiment has to be repeated with additional random seeds as well as on other tasks in different languages to check whether our findings can be confirmed. There is still a lot to investigate, but this work provides the initial basis for fruitful further research and for the first time focuses on the initial acquisition model in a DAL setting.

As for our task, we will expand it to examples of different categories (listed in 3.1) as in the error analysis (see Section 4.2) we learnt that the trained model detects them. As we are interested in these anyway, this is very practical. With the aid of a text classification model (that is yet to be trained) or a rule-based approach we can then tag the examples with the corresponding categories.

Limitations

There is no state-of-the-art approach and we cannot (afford) to annotate the whole unlabeled data pool: therefore we do not know how the model would perform on the whole dataset.

As we only analysed the impact of the acquisition models for one task and one language it remains to be seen how scalable the findings are.

As only a limited number of random seeds are considered for the initial acquisition model, it is impossible to determine if a model exists that performs worse than the already identified worst initial acquisition model.

Acknowledgement

This work was performed on the computational resource bwUniCluster funded by the Ministry of Science, Research and the Arts Baden-Württemberg and the Universities of the State of Baden-Württemberg, Germany, within the framework program bwHPC.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. 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)*, pages 54–59, Minneapolis, Minnesota. Association for Computational Linguistics.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649.

Eva Breindl. 2016. Sprachberatung im interaktiven Web. In Staffeldt Sven Klein Wolf Peter, editor, *Die Kodifizierung der Sprache. Strukturen, Funktionen, Konsequenzen*, volume 17 of *WespA - Würzburger elektronische sprachwissenschaftliche Arbeiten*, pages 85–109.

Aditi Chaudhary, Antonios Anastasopoulos, Zaid Sheikh, and Graham Neubig. 2021. Reducing confusion in active learning for part-of-speech tagging. *Transactions of the Association for Computational Linguistics*, 9:1–16.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Mengyi Gao, Canran Xu, and Peng Shi. 2021. Hierarchical Character Tagger for Short Text Spelling Error Correction. In *Proceedings of the Seventh Workshop on Noisy User-generated Text (W-NUT 2021)*, pages 106–113, Online. Association for Computational Linguistics.

Yekyung Kim. 2020. Deep Active Learning for Sequence Labeling Based on Diversity and Uncertainty in Gradient. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 1–8, Suzhou, China. Association for Computational Linguistics.

Christian Lang, Ngoc Duyen Tanja Tu, Roman Schneider, and Anna Volodina. 2023a. Projektvorstellung – Sprachanfragen. Empirisch gestützte Erforschung von Zweifelsfällen. In *9. Tagung des Verbands "Digital Humanities im deutschsprachigen Raum" DHd 2023*, Trier, Luxemburg.

Christian Lang, Ngoc Duyen Tanja Tu, and Laura Zeidler. 2023b. Making Non-Normalized Content Retrievable – A Tagging Pipeline for a Corpus of Expert–Layperson Texts. In *Proceedings of the 4th Conference on Language, Data and Knowledge*, pages 239–244, Portugal. NOVA CLUNL.

David D. Lewis and William A. Gale. 1994. A Sequential Algorithm for Training Text Classifiers. Publisher: arXiv Version Number: 2.

Warren S. McCulloch and Walter Pitts. 1943. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133.

Seyed Morteza Mirbostani, Yasaman Boreshban, Salam Khalifa, SeyedAbolghasem Mirroshandel, and Owen Rambow. 2023. Deep Active Learning for Morphophonological Processing. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 793–803, Toronto, Canada. Association for Computational Linguistics.

Puria Radmard, Yassir Fathullah, and Aldo Lipani. 2021. Subsequence Based Deep Active Learning for Named Entity Recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4310–4321, Online. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2017. Reporting Score Distributions Makes a Difference: Performance Study of LSTM-networks for Sequence Tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 338–348, Copenhagen, Denmark. Association for Computational Linguistics.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B. Gupta, Xiaojiang Chen, and Xin Wang. 2022. A Survey of Deep Active Learning. *ACM Computing Surveys*, 54(9):1–40.

Artem Shelmanov, Dmitri Puzyrev, Lyubov Kupriyanova, Denis Belyakov, Daniil Larionov, Nikita Khromov, Olga Kozlova, Ekaterina Artemova, Dmitry V. Dylov, and Alexander Panchenko. 2021. Active Learning for Sequence Tagging with Deep Pre-trained Models and Bayesian Uncertainty Estimates. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1698–1712, Online. Association for Computational Linguistics.

Yanyao Shen, Hyokun Yun, Zachary Lipton, Yakov Kronrod, and Animashree Anandkumar. 2017. Deep Active Learning for Named Entity Recognition. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 252–256, Vancouver, Canada. Association for Computational Linguistics.

Akim Tsvigun, Artem Shelmanov, Gleb Kuzmin, Leonid Sanochkin, Daniil Larionov, Gleb Gusev, Manvel Avetisian, and Leonid Zhukov. 2022. Towards Computationally Feasible Deep Active Learning. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1198–1218, Seattle, United States. Association for Computational Linguistics.

Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2022. A Survey of Active Learning for Natural Language Processing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6166–6190, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.