



Data Science & Analytics
Center of Excellence

DEEP ACTIVE LEARNING FOR CLINICAL ENTITY RECOGNITION

Nadia Abraham

www.Geninvo.com
North America | UK | Europe | Asia



Contents

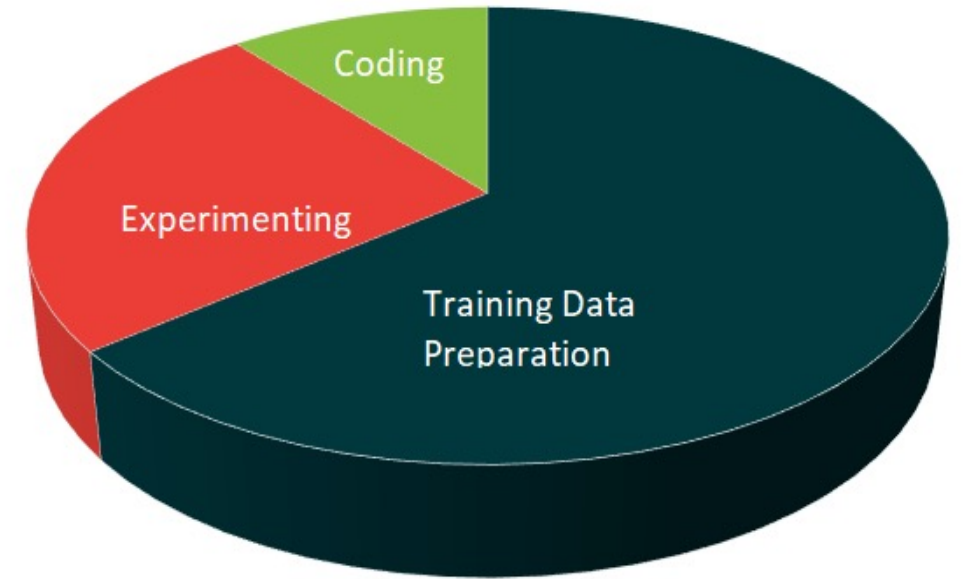
- ▶ Introduction
- ▶ Named Entity recognition
- ▶ Active Learning “philosophy”
- ▶ Active learning scenarios
- ▶ Pool-Based Sampling: pseudo algorithm
- ▶ Acquisition functions
- ▶ Our Experiment
- ▶ Dataset Exploration
- ▶ NER using simpletransformers and BioBERT
- ▶ Evaluation Metrics
- ▶ Active Learning loop
- ▶ Results and Inferences
- ▶ End Notes
- ▶ References



www.geninvo.com
North America | UK | Europe | India

Introduction : Key Data challenges in Deep learning

- ▶ Big picture: more Data — smarter AI
- ▶ “Unlabeled” Data is everywhere
- ▶ Lack of sufficient labeled data
- ▶ Lots of “valuable” effort spend on annotating data
- ▶ Motivation for **Active learning**



presentation and subsequent type two diabetes mellitus PROBLEM (T2DM PROBLEM), one prior episode of HTG-induced pancreatitis PROBLEM three years prior to presentation , associated with an acute hepatitis PROBLEM , and obesity PROBLEM with a body mass index TEST (BMI TEST) of 33.5 kg/m2 , presented with a one-week history of polyuria PROBLEM , polydipsia PROBLEM , poor appetite PROBLEM , and vomiting PROBLEM . Two weeks prior to presentation , she was treated with a five-day course of amoxicillin TREATMENT . She was on metformin TREATMENT .

Objective

Attempt to demystify the concept of **active learning** in AI and propose an approach to apply it to enhance deep learning models in context of **clinical entity recognition**.



GENINVO™

www.geninvo.com
North America | UK | Europe | India

Our Deep learning task: Named Entity recognition

Named entity recognition (NER) , also known as entity chunking/extraction , is a popular technique used in information retrieval to identify and segment the named entities and classify or categorize them under various predefined classes.

NER in Clinical/Biomedical Domain

Ever-growing unstructured data footprint in clinical domain

Data sources - EHR/EMRs, CRFs, bioresearch data, pathological reports, clinical narratives etc.

Identify clinically relevant concepts from free clinical text

Conversion to structured representations

Supporting important use cases such as disease phenotyping, patient cohort identification, decision support, and drug repurposing.



GENINVO™

www.geninvo.com
North America | UK | Europe | India

Active Learning “philosophy”

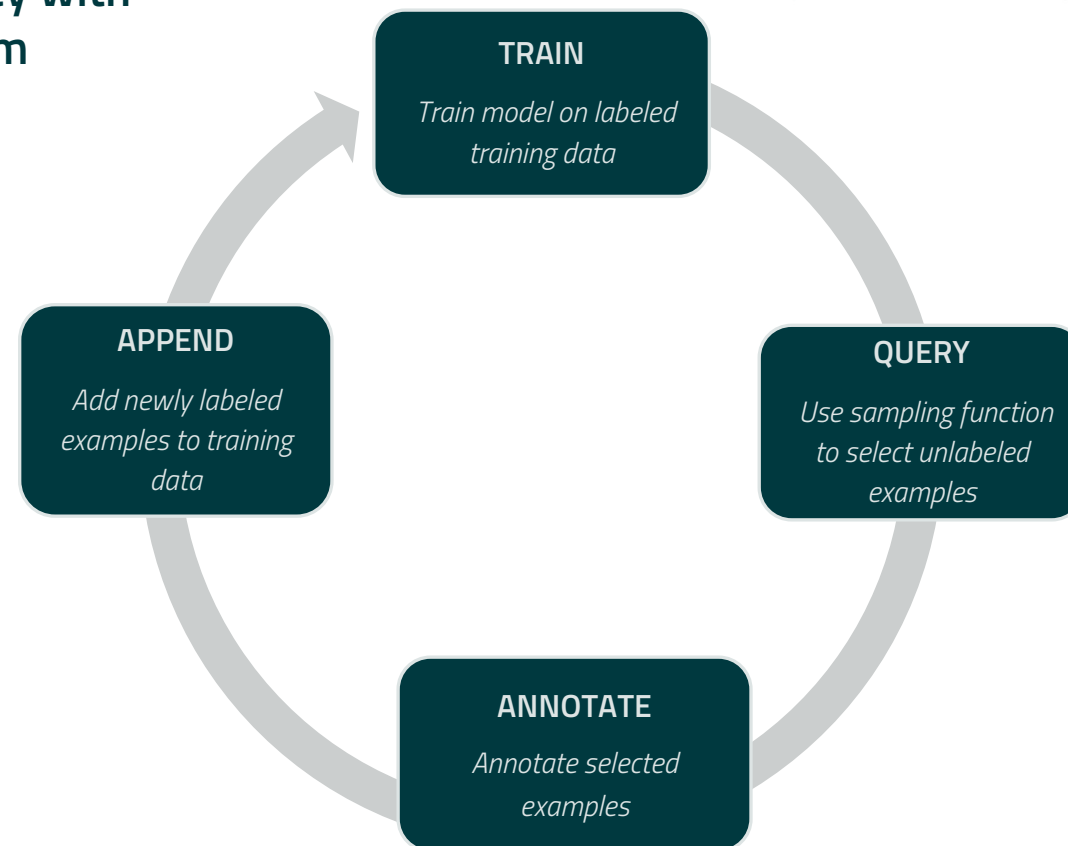
“a machine learning algorithm can achieve greater accuracy with fewer training labels if it is allowed to choose the data from which it learns”

Semi-supervised learning

Sample selection - Aims to select the most informative samples from the unlabeled_dataset and pass it on to the annotators(oracle) for labelling.

Iterative and Incremental approach

Significant reduction in annotation cost



General Steps

1. A small sample of data must be selected from the unlabeled data for manual tagging.

2. Once the data is tagged, it must be used to train the model. The initial model will provide insight into which areas of the parameter space should be annotated first.

3. Once the model is trained, it should be used to make prediction on the unlabeled data.

4. A prioritization score is calculated for each unlabeled data point based on the prediction of the model. Based on the prioritization score, data points can be sampled from the unlabeled data, annotated and appended to training data.



Active learning scenarios

Membership Query

Synthesis

The learner generates its own instance from an underlying natural distribution. For example, if the dataset are pictures of humans and animals, the learner could send a clipped image of a leg to the oracle and query if this appendage belongs to an animal or human. This is particularly useful if your dataset is small.

Stream-Based Selective Sampling

Each unlabeled data point is examined one at a time with the machine evaluating the informativeness of each item against its query parameters. The learner decides for itself whether to assign a label or query the oracle for each datapoint.

Pool-Based Sampling

In this scenario, instances are drawn from the entire data pool and assigned an informative score, a measurement of how well the learner “understands” the data. The system then selects the most informative instances and queries the oracle for the labels.



GENINVO™

www.geninvo.com
North America | UK | Europe | India

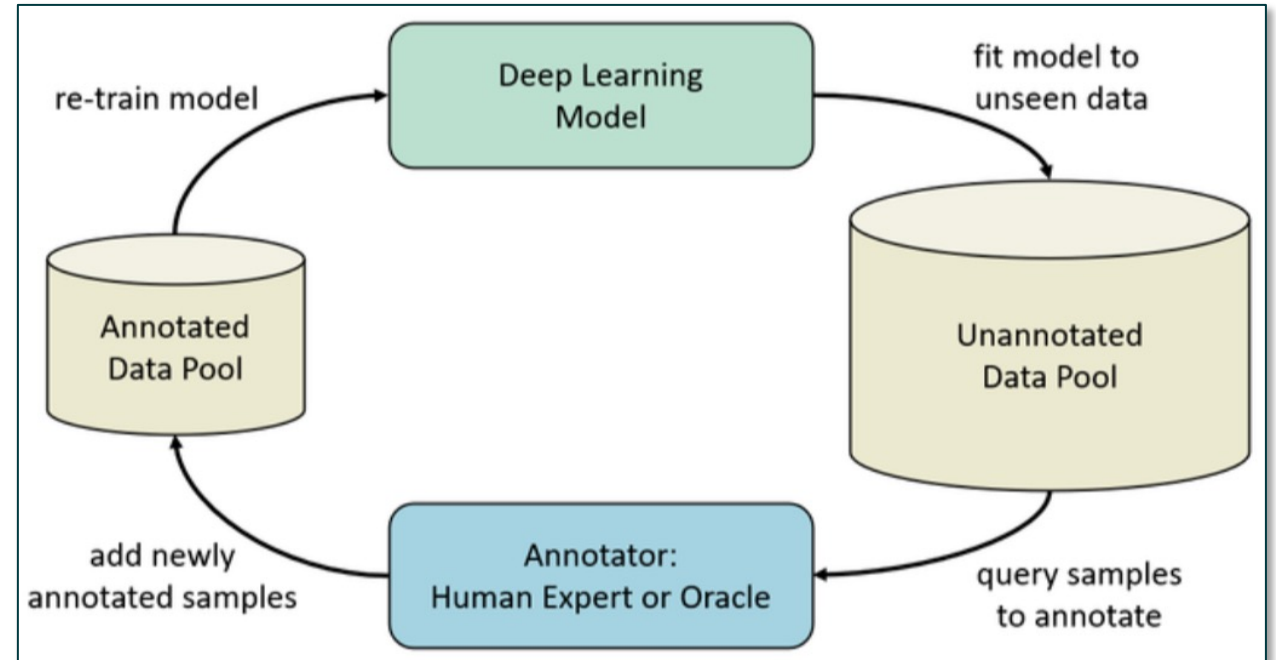
Pool-Based Sampling: pseudo algorithm

- ▶ Divide data into un-labelled pool P and test set S

Iterate {

- ▶ Randomly select k examples from training pool P and add to initialized training set T
- ▶ Train the model using T
- ▶ Use the training model with test set to get performance measures
- ▶ Rank and select k most informative samples from pool P based on uncertainty metric
- ▶ Annotate and move these k examples to training set T
- ▶ Remove these k examples from pool P

}



Acquisition functions

Least Confidence sampling – difference between the most confident prediction and 100% confidence. **Confidence** of a prediction is obtained by converting the raw model output to a 0–1 range of scores using *softmax*.

The samples with least confidence is selected for next iteration of training

$$\sigma(x_j) = \frac{e^{x_j}}{\sum_i e^{x_i}}$$

```
from scipy.special import softmax
```

Maximum Entropy sampling - difference between all predictions, as defined by information theory. **Entropy** - In information theory, the entropy of a random variable is the average level of “information”, “surprise”, or “uncertainty” inherent in the variable’s possible outcomes.

The samples with maximum entropy is selected for next iteration of training

$$H(X) = - \sum_{i=1}^n P(x_i) \log P(x_i)$$

```
import numpy as np
def entropy(l):
    return -sum([x * np.log(np.clip(x, 1e-12, 1 - 1e-12)) for x in l])
```



GENINVO™

www.geninvo.com
North America | UK | Europe | India

Our Experiment

Uncertainty-based heuristic
for active learning with
Python `simpletransformers`
and BC5CDR dataset.

BC5CDR Dataset – NER tagged
Chemical and Disease entities

Active
Learning

Data pre-processing
*Tokenization, Sentence ID
generation*

`simpletransformers`
NER Model
Contextual Embedding
*Pre-Trained model -
BioBERT*

Output
Named Entities

Evaluation

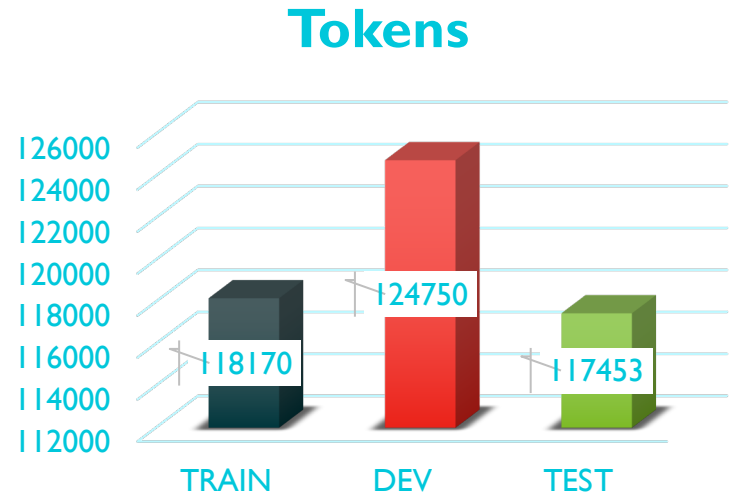
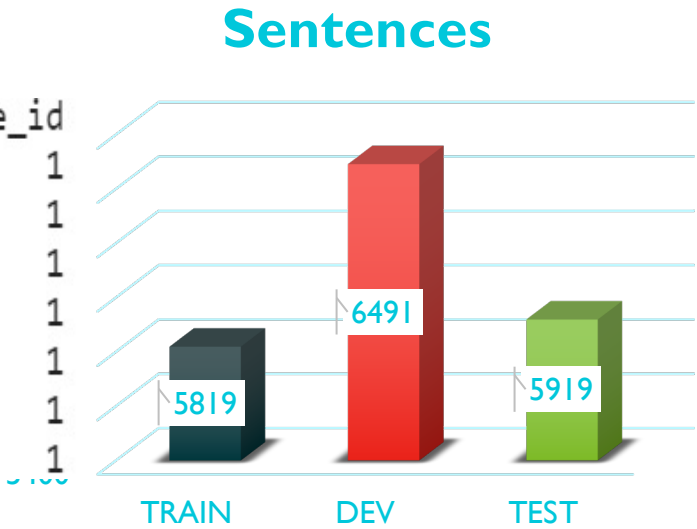


www.geninvo.com
North America | UK | Europe | India

BC5CDR: Dataset Exploration

Consists of 1500 PubMed articles with 4409 annotated chemicals, 5818 diseases and 3116 chemical-disease interactions in IOB format tags (short for inside, outside, beginning)

words	labels	sentence_id
22	B-Chemical	1
-	I-Chemical	1
oxacalcitriol	I-Chemical	1
suppresses	0	1
secondary	B-Disease	1
hyperparathyroidism	I-Disease	1
without	0	1



NER using simpletransformers and BioBERT

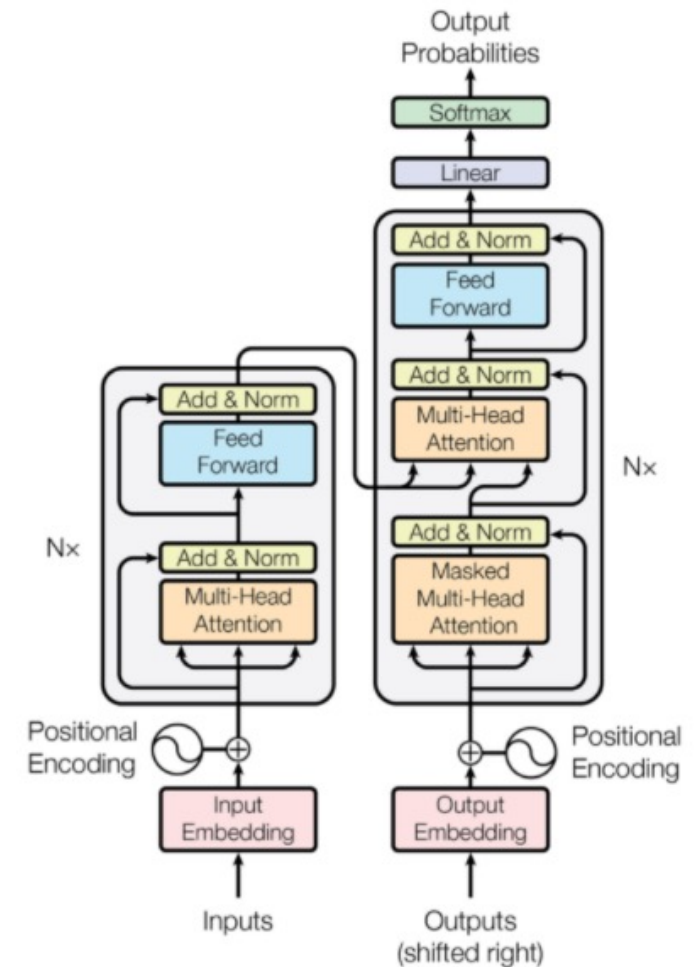
Transformer is a SOTA architecture that aims to solve sequence-to-sequence tasks (like NER) while handling long-range dependencies with ease.

Key Concepts

- **Attention** is the mapping of a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The attention mechanism allows the model to understand the context of a text.
- **Multi-head attention** allows the model to jointly attend to information from different representation subspaces at different positions.

Basic Architecture

- The **Encoder** block on the left has 1 layer of a Multi-Head Attention followed by another layer of Feed Forward Neural Network.
- The **Decoder**, on the other hand, has an extra Masked Multi-Head Attention.
- The encoder and decoder blocks are **multiple identical encoders and decoders** stacked on top of each other.



simpletransformers

simpletransformers is a Natural Language Processing (NLP) library designed to simplify the usage of Transformer models without having to compromise on utility.

Key Features:

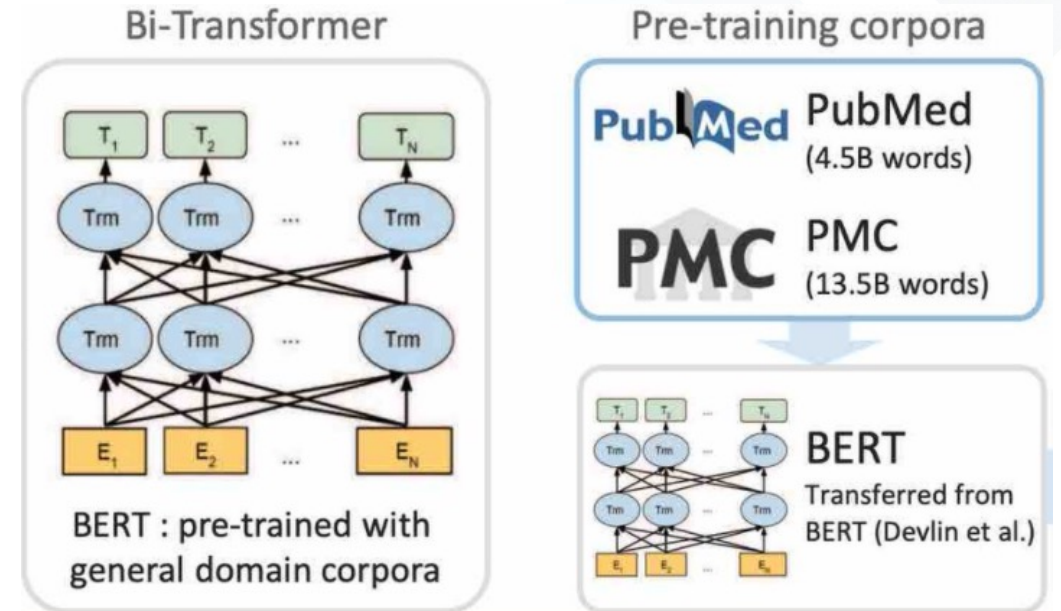
- ▶ Allows to quickly train and evaluate transformer models.
- ▶ Supports Sequence Classification, Token Classification (NER), Question Answering, Language Model Fine-Tuning, Language Model Training, Language Generation, T5 Model, Seq2Seq Tasks , Multi-Modal Classification and Conversational AI.
- ▶ Provides separate class for every NLP task.
- ▶ Input data formats are optimized for the task
- ▶ Outputs are clean and ready-to-use for the task with minimal to no post-processing required
- ▶ In our experiment, we use **NERModel** class from **simpletransformers.ner**



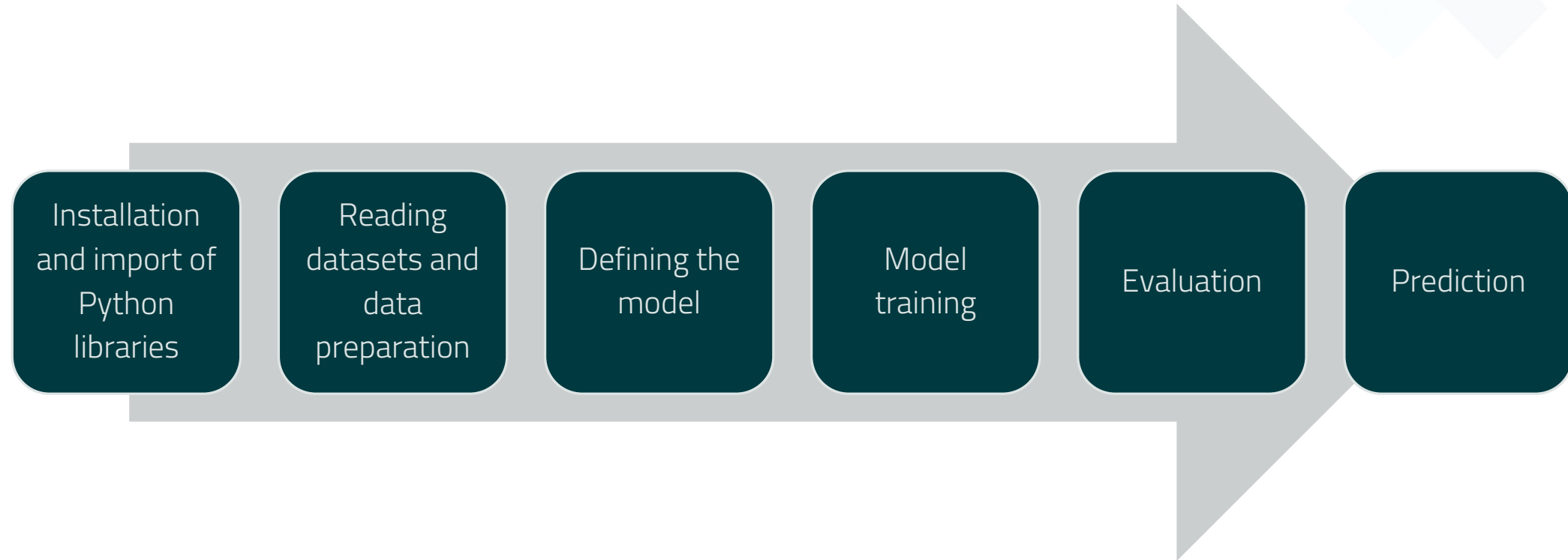
www.geninvo.com
North America | UK | Europe | India

BioBERT

- ▶ BioBERT (Bidirectional Encoder Representations from Transformers for Biomedical Text Mining) is a domain-specific language representation model pre-trained on large-scale biomedical corpora.
- ▶ Large pre-trained model that can be used in a transfer learning setting for biomedical text mining.
- ▶ Trained with a BERT pre-trained model trained on Wikipedia 2.5 billion words & Books Corpus 0.8 billion words and PubMed Abstracts 4.5 billion words & PMC Full-text articles 13.5 billion words.



NER Model – Process flow



NER Model – Code in detail

Installation and import of Python libraries

```
!pip install -q transformers  
!pip install -q simpletransformers
```

```
from simpletransformers.ner import NERModel  
from transformers import AutoTokenizer
```



www.geninvo.com
North America | UK | Europe | India

Reading datasets and data preparation

```
import pandas as pd
def read_input(filename):
    df = pd.read_csv(filename,
                      sep = '\t', header = None, keep_default_na = False,
                      names = ['words', 'labels'],
                      quoting = 3, skip_blank_lines = False)

    # Remove the -DOCSTART- header
    df = df[~df['words'].astype(str).str.startswith('-DOCSTART- ')]
    id_no = list()
    counter=1
    for index,word in enumerate(df['words']):
        id_no.append(counter)
        if word == '.':
            counter+=1
    df['sentence_id'] = id_no
    return df[df.words != '']
```

```
train_df = read_input("/input/bc5cdr/train.tsv")
test_df = read_input("/input/bc5cdr/test.tsv")
dev_df = read_input("/input/bc5cdr/devel.tsv")
```



Defining the model

```
train_args = {
    'reprocess_input_data': True,
    'overwrite_output_dir': True,
    'sliding_window': True,
    'max_seq_length': 64,
    'num_train_epochs': 10,
    'train_batch_size': 32,
    'fp16': True,
    'output_dir': '/outputs/',
    'best_model_dir': '/outputs/best_model/',
    'evaluate_during_training': True,
    'manual_seed': 55
}

custom_labels = list(train_df['labels'].unique())

['B-Chemical', 'O', 'B-Disease', 'I-Disease', 'I-Chemical']

model = NERModel('bert', 'dmis-lab/biobert-v1.1', labels=custom_labels, args=train_args,
                 use_cuda=True)
```

Model training

```
model.train_model(train_df, eval_data=dev_df, verbose=False)
```

100%  3/3 [00:07<00:00, 7.02s/it]
Epoch 1 of 1: 100%  1/1 [01:24<00:00, 84.51s/it]
Epochs 0/1. Running Loss: 0.0740: 100%  182/182 [00:46<00:00, 3.58it/s]
100%  3/3 [00:07<00:00, 7.56s/it]
Running Evaluation: 100%  740/740 [00:18<00:00, 39.64it/s]
100%  3/3 [00:07<00:00, 7.82s/it]
Running Evaluation: 100%  812/812 [00:20<00:00, 39.74it/s]

Evaluation

```
result, model_outputs, preds_list = model.eval_model(test_df, verbose=False)
```

result – default metrics: cross entropy evaluation loss, precision, recall, f1 score

```
{'eval_loss': 0.07658909394853379, 'precision': 0.8278928850664582, 'recall': 0.8830397164599187, 'f1_score': 0.854577553593947}
```

model_outputs – raw model outputs

preds_list – List of predictions



GENINVO™

www.geninvo.com
North America | UK | Europe | India

Evaluation Metrics explained

Basics:

- ▶ True Positive (TP): Model and Ground Truth Match
- ▶ False Positive (FP): The entities Model identifies that are wrong in Ground Truth
- ▶ False Negative (FN): The entities Model misses that are in the Ground Truth.

- ▶ Accuracy = $(TP+TN)/(TP+FP+TN+FN)$ - the number of correctly categorized examples divided by the total number of examples
- ▶ Precision = $TP/(TP+FP)$ which describes the percentage of correct entities found by the NER model.
- ▶ Recall = $TP/(TP+FN)$ which describes the percentage of ground truth entities that are retrieved from the NER model.
- ▶ F1 Score = $2 * Precision * Recall / (Precision + Recall)$ which is the harmonic mean of precision and recall.

Evaluation Metrics explained (Cont'd)

- ▶ Accuracy is relatively easier to understand but it does not consider the subtleties of class imbalances and hence not suitable for imbalanced data tasks like NER.
- ▶ A poor precision means the NER model may identify a lot of the entities but many of them are not really in the ground truth, while a poor recall means many of the entities in the ground truth are not discovered by the NER model. F1 score balances both precision and recall.
- ▶ F1 Score is generally preferred over Accuracy in Information Retrieval tasks.

Model Prediction

```
model.predict("Torsade de pointes ventricular tachycardia during\n",
              " low dose intermittent dobutamine treatment in a\n",
              " patient with dilated cardiomyopathy and\n",
              " congestive heart failure.")
sample = test_df[test_df.sentence_id == 1].words.str.cat(sep=' ')
samples = [sample]
predictions, _ = model.predict(samples)
for idx, sample in enumerate(samples):
    print('{}: '.format(idx))
    for word in predictions[idx]:
        print('{} '.format(word))
```

```
0:
{'Torsade': 'B-Disease'}
{'de': 'I-Disease'}
{'pointes': 'I-Disease'}
{'ventricular': 'B-Disease'}
{'tachycardia': 'I-Disease'}
{'during': 'O'}
{'low': 'O'}
{'dose': 'O'}
{'intermittent': 'O'}
{'dobutamine': 'B-Chemical'}
{'treatment': 'O'}
{'in': 'O'}
{'a': 'O'}
{'patient': 'O'}
{'with': 'O'}
{'dilated': 'B-Disease'}
{'cardiomyopathy': 'I-Disease'}
{'and': 'O'}
{'congestive': 'B-Disease'}
{'heart': 'I-Disease'}
{'failure': 'I-Disease'}
{'.': 'O'}
```



GENINVO™

www.geninvo.com
North America | UK | Europe | India

Active Learning loop – code excerpts

```
# Least confidence strategy
if strategy == "LC":
    for sent_score in scores:
        for word_score in sent_score:
            word_conf = softmax(word_score)
            word_min_conf = min(word_conf)
            word_least_conf_list.append(word_min_conf)
        sent_least_conf = min(word_least_conf_list)
        sent_least_conf_list.append(sent_least_conf)
    scores_sent = sent_least_conf_list
    scores_sent = [1- elem for elem in scores_sent]

#Maximum Entropy strategy
elif strategy == "ME":
    for sent_score in scores:
        for word_score in sent_score:
            word_conf = softmax(word_score)
            word_entr = entropy(word_conf)
            word_entr_list.append(word_entr)
        sent_max_entr = max(word_entr_list)
        sent_max_entr_list.append(sent_max_entr)
    scores_sent = sent_max_entr_list
```

Calculation of score per sentence based on the strategy selected



GENINVO™

www.geninvo.com
North America | UK | Europe | India

Active Learning loop – code excerpts (Cont'd)

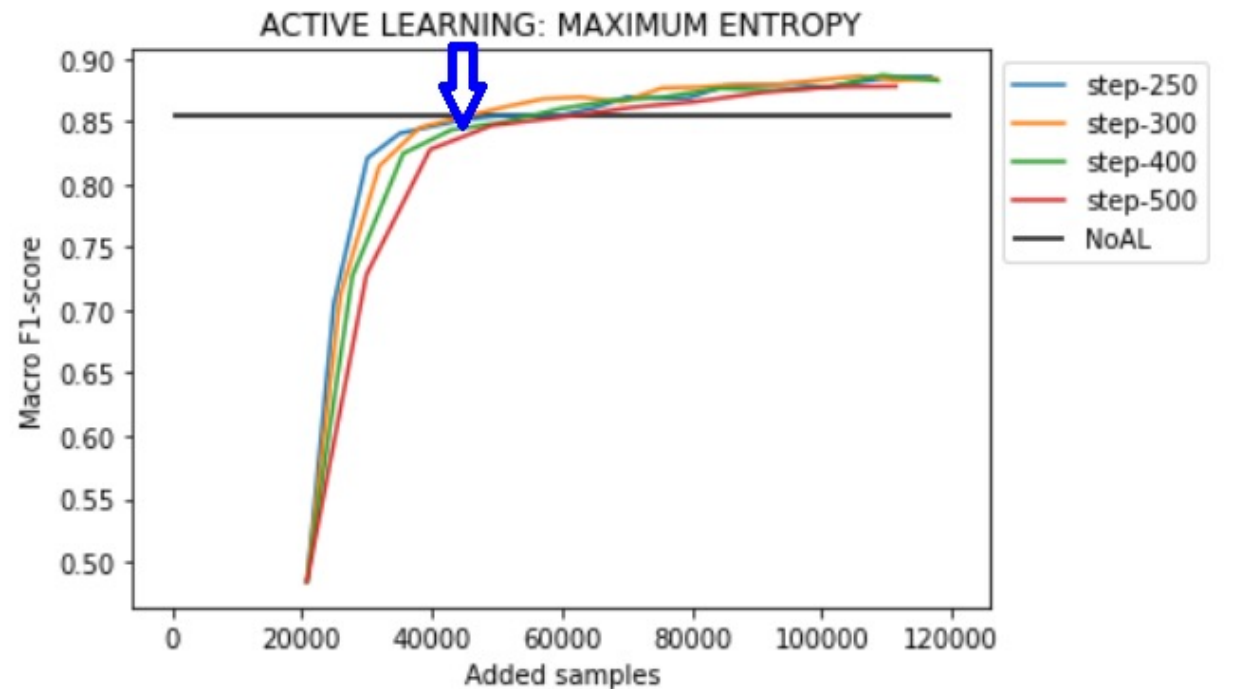
The next step of samples based on the scores is selected from the pool and sent for annotation. These samples are then added to the dataset used for model training.

```
dfs = [x for _, x in left.groupby('sentence_id')]
print("dfs count: ", len(dfs))
step_samples_zip = sorted(zip(dfs, scores_sent), key = lambda x: x[1], reverse=True)[0:step]
step_samples = [x for x, v in step_samples_zip]
print("step_samples count: ", len(step_samples))
for sample in step_samples:
    selected = selected.append(sample)

left = left[~left.sentence_id.isin(selected['sentence_id'])]
```

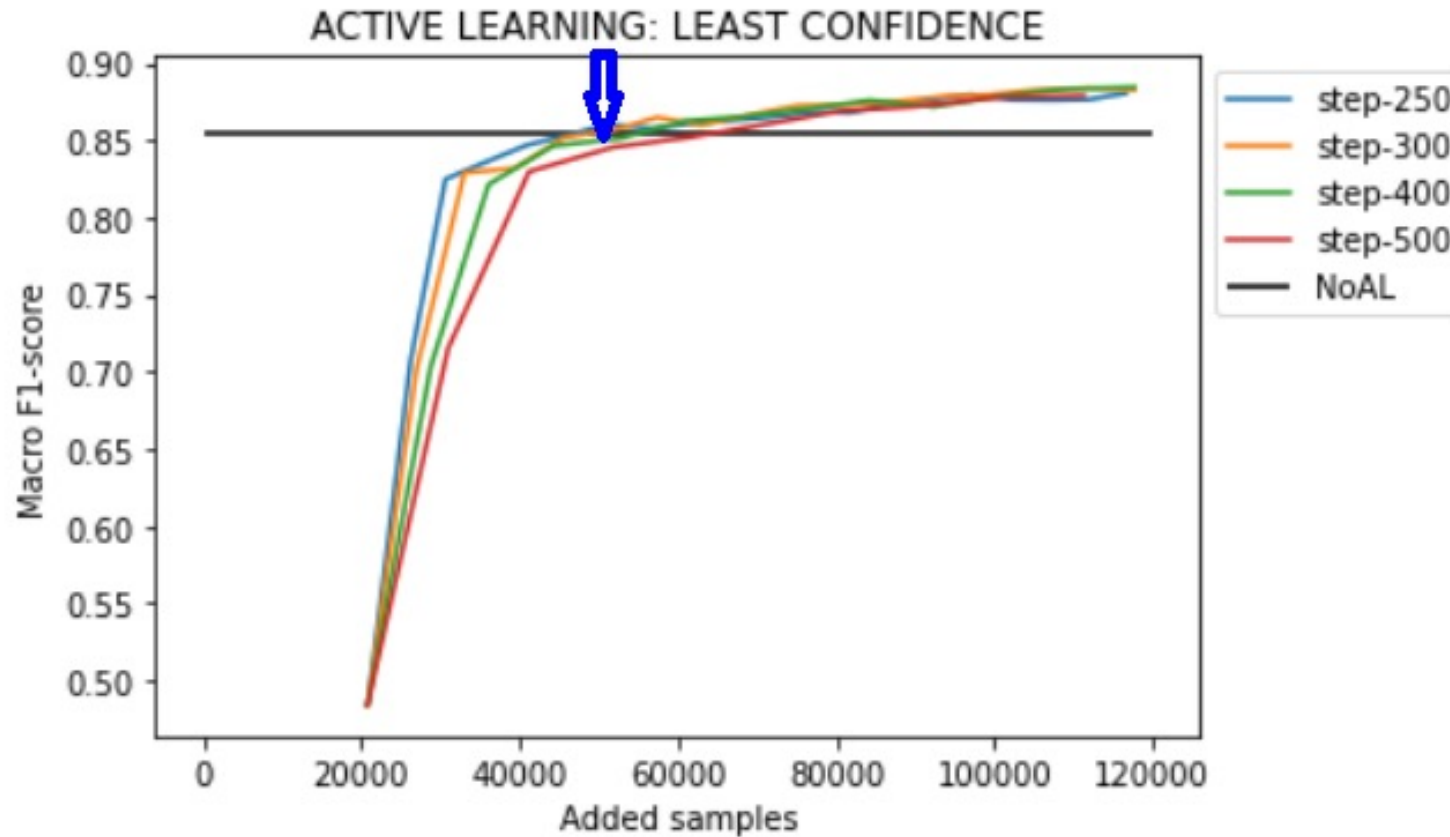
Results and Inferences

In our experiment, we have tried with the following step sizes: 250, 300, 400 and 500. For example, step size =250 means an increment of 250 sentences will be fed to the model in an iteration.



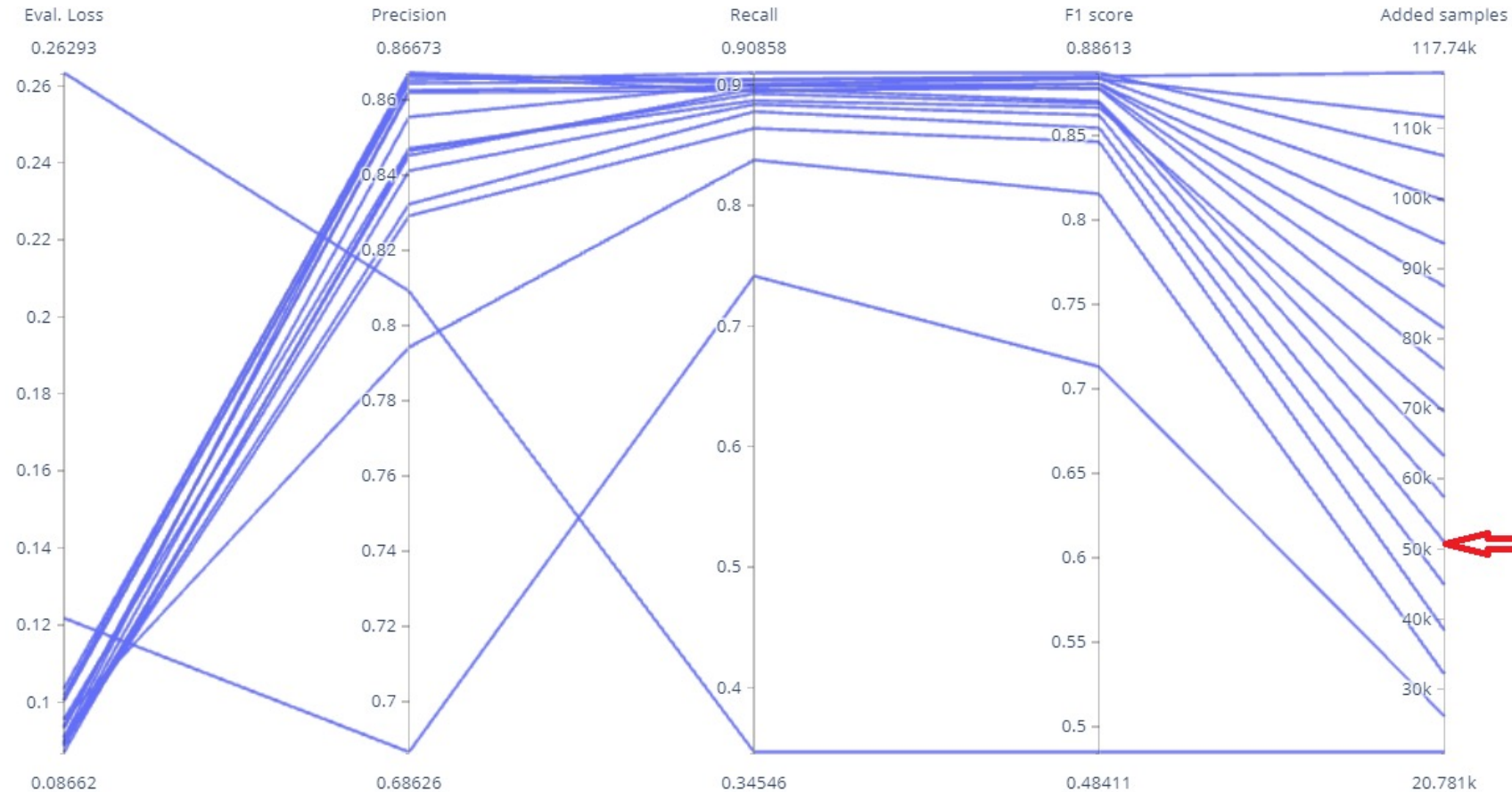
- ▶ Optimum Step: step=300
- ▶ Reaches NoAL F1 score with 50777 samples.
- ▶ Maximum F1 score - 0.8840

Results and Inferences (Cont'd)



- ▶ Optimum Step: step=300
- ▶ Reaches NoAL F1 score with 51881 samples
- ▶ Maximum F1 score - 0.8825

Results and Inferences (Cont'd)



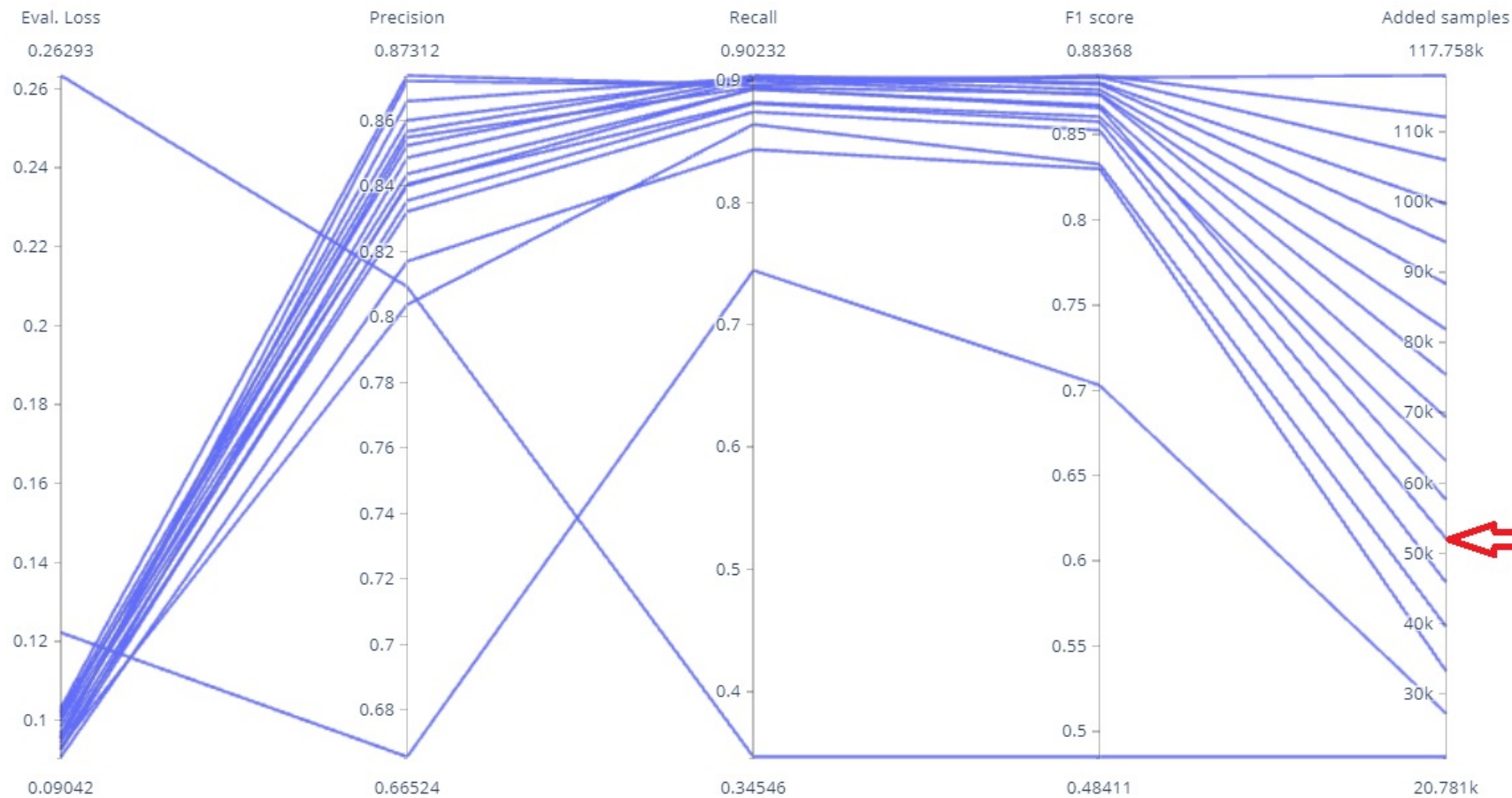
Maximum entropy
step=300



www.geninvo.com
North America | UK | Europe | India

Parallel Coordinate Plots

Results and Inferences (Cont'd)



Least confidence
step=300

End Notes

In this paper, we have proposed deep active learning algorithms for Clinical data named entity recognition and empirically demonstrated that using effective active learning techniques requires **less than half the labelled data** to achieve the same performance as a normal deep learning model without active learning.

- ▶ Deep Active Learning leverages the common advantages of DL and AL
 - ▶ DL's ability to process high-dimensional data and conduct automatic feature extraction
 - ▶ AL's potential to effectively reduce annotation costs.
- ▶ Especially useful in areas where labels require high levels of expertise



www.geninvo.com
North America | UK | Europe | India

GENINVO™

End Notes (Cont'd)

Challenges:

- ▶ Same DAL method may yield different results in different tasks - task independence is an important research direction
- ▶ Softmax response in deep learning is not often a reliable measure of confidence – may impact performance

Future Research:

- ▶ Explore other sample selection strategies
- ▶ Extend to other SOTA models
- ▶ Micro F1 score per entity for evaluation



www.geninvo.com
North America | UK | Europe | India

References

- ▶ <https://deepai.org/>
- ▶ [Active learning — Uncertainty Sampling \(P3\) | by Duy Anh Nguyen | Medium](#)
- ▶ <https://academic.oup.com/database/article/doi/10.1093/database/baw068/2630414>
- ▶ <http://www.crownpku.com/2020/03/02/Evaluation-of-Named-Entity-Recognition-Model.html>
- ▶ [Active Learning Tutorial. An active learning performance... | by Ori Cohen | Towards Data Science](#)
- ▶ [Deep Active Learning for Named Entity Recognition \(aclanthology.org\)](#)
- ▶ [NER Model - Simple Transformers](#)
- ▶ [BioBERT: a pre-trained biomedical language representation model for biomedical text mining - PubMed \(nih.gov\)](#)



GENINVO™

www.geninvo.com
North America | UK | Europe | India



Ideas Delivered Intuitively

Thank You!

nadia.a25@geninvo.com | www.geninvo.com

