Build a Customized NLP Service

Interim Presentation

Friday, July 9, 2021

Presented by: Yongchao Zhou

VECTOR INSTITUTE

Motivation

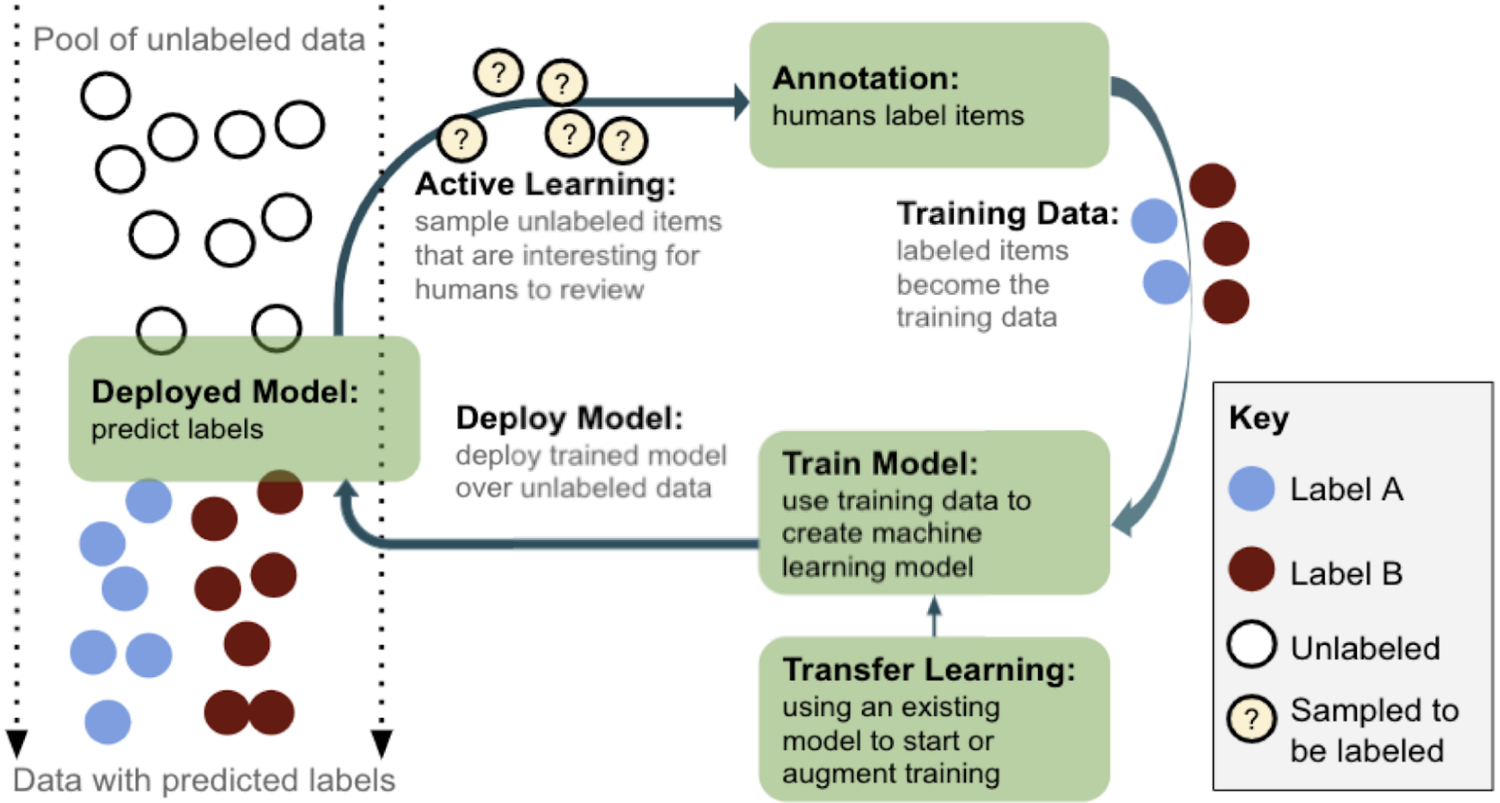


Figure 1: Machine Learning Life Cycle

- ML Application Life Cycle
  - Data Collection
  - Data Annotation
  - Model Training & Evaluation
  - Model Deployment
- ML Tools
  - Active Learning
  - Transfer Learning

Motivation

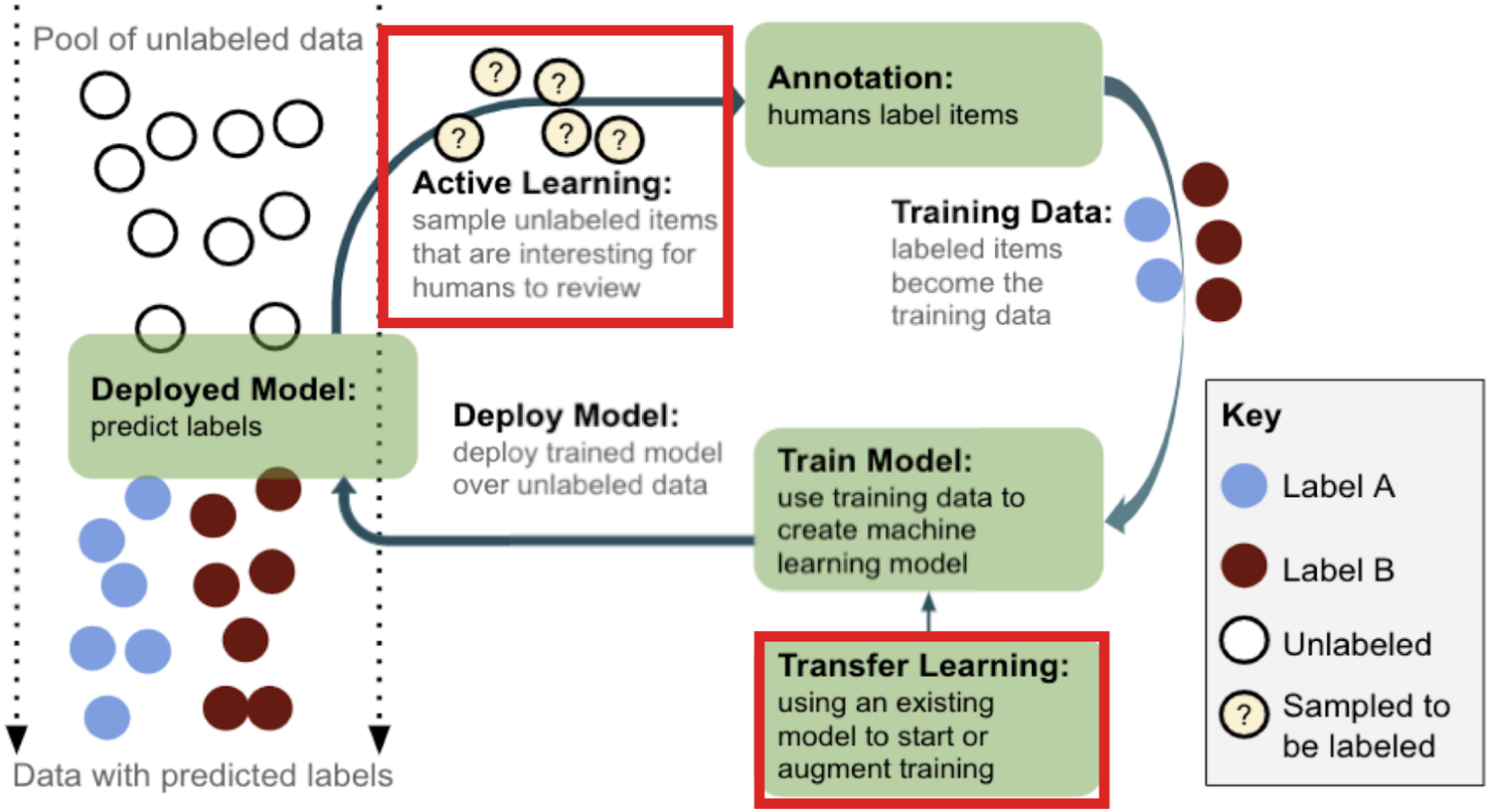


Figure 1: Machine Learning Life Cycle

- ML Application Life Cycle
  - Data Collection
  - Data Annotation
  - Model Training & Evaluation
  - Model Deployment
- ML Tools
  - Active Learning
  - Transfer Learning

NER service

- Inference API
- Graphical User Interface

NER service

- Inference API

```
>>> import NerModel
>>> model = NerModel(args.model_dir)
>>> model(args.text)

{'text': 'Geoffrey Everest Hinton (born 6 December 1947)
'ents': [{'token': 'Geoffrey Everest Hinton', 'start': 0,
        {'token': '6', 'start': 30, 'end': 31, 'label':
        {'token': 'December 1947', 'start': 32, 'end': 45
        {'token': 'British', 'start': 52, 'end': 59, 'lab
```

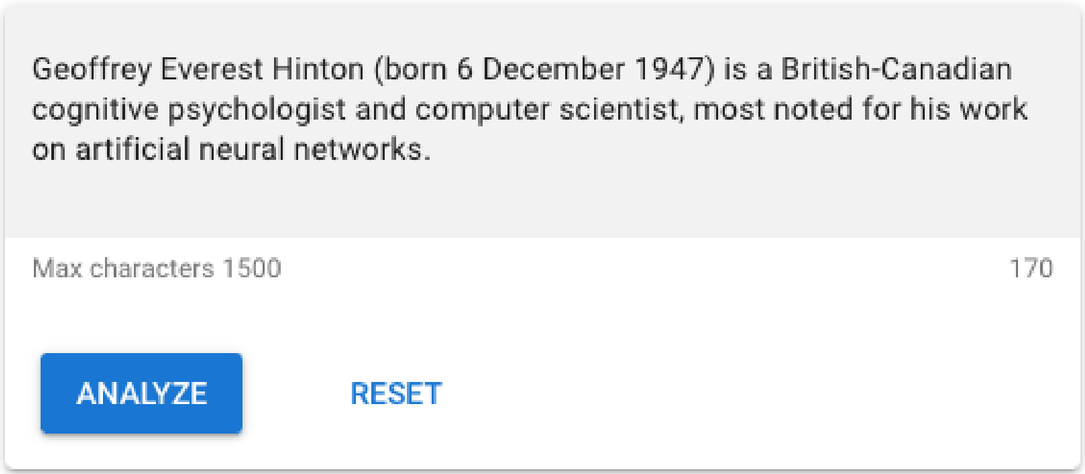
- Graphical User Interface

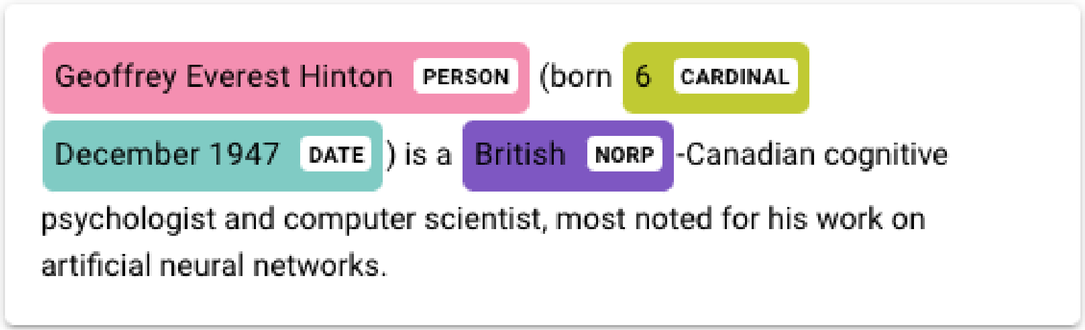
NER service

- Inference API

```
>>> import NerModel
>>> model = NerModel(args.model_dir)
>>> model(args.text)

{'text': 'Geoffrey Everest Hinton (born 6 December 1947)
'ents': [{'token': 'Geoffrey Everest Hinton', 'start': 0,
        {'token': '6', 'start': 30, 'end': 31, 'label':
        {'token': 'December 1947', 'start': 32, 'end': 45
        {'token': 'British', 'start': 52, 'end': 59, 'lab
```

- Graphical User Interface





How to build a customized NER service?

- Data Collection -> Data Annotation -> Model Training & Evaluation -> Model Deployment (API/GUI)

How to build a customized NER service?

- Data Collection -> Data Annotation -> Model Training & Evaluation -> Model Deployment (API/GUI)

How to build a customized NER service?

- Data Collection -> Data Annotation -> Model Training & Evaluation -> Model Deployment (API/GUI)

```
>>> raw_dataset[0]
"Geoffrey Everest Hinton (born 6 December 1947) is a British-Canadian cognitive psychologist and computer scientist, mos
```

```
>>> train_dataset[0]
{'text': 'Geoffrey Everest Hinton (born 6 December 1947) is a British-Canadian cognitive psychologist and computer scier
'annotation': [{'token': 'Geoffrey', 'label': 'PERSON', 'iob': 3}, {'token': 'Everest', 'label': 'PERSON', 'iob': 1}, {'
        {'token': 'born', 'label': 'null', 'iob': 2}, {'token': '6', 'label': 'CARDINAL', 'iob': 3}, {'token': 'Decembe
        {'token': ')', 'label': 'null', 'iob': 2}, {'token': 'is', 'label': 'null', 'iob': 2}, {'token': 'a', 'label':
        {'token': '-', 'label': 'null', 'iob': 2}, {'token': 'Canadian', 'label': 'null', 'iob': 2}, {'token': 'cogniti
        {'token': 'and', 'label': 'null', 'iob': 2}, {'token': 'computer', 'label': 'null', 'iob': 2}, {'token': 'scier
        {'token': 'most', 'label': 'null', 'iob': 2}, {'token': 'noted', 'label': 'null', 'iob': 2}, {'token': 'for', '
        {'token': 'work', 'label': 'null', 'iob': 2}, {'token': 'on', 'label': 'null', 'iob': 2}, {'token': 'artificial
        {'token': 'networks', 'label': 'null', 'iob': 2}, {'token': '.', 'label': 'null', 'iob': 2}
        ... ]}
```

How to build a customized NER service?

- Data Collection -> Data Annotation -> Model Training & Evaluation -> Model Deployment (API/GUI)

```
>>> raw_dataset[0]
"Geoffrey Everest Hinton (born 6 December 1947) is a British-Canadian cognitive psychologist and computer scientist, mos
```

```
>>> train_dataset[0]
{'text': 'Geoffrey Everest Hinton (born 6 December 1947) is a British-Canadian cognitive psychologist and computer scier
'annotation': [{'token': 'Geoffrey', 'label': 'PERSON', 'iob': 3}, {'token': 'Everest', 'label': 'PERSON', 'iob': 1}, {'
        {'token': 'born', 'label': 'null', 'iob': 2}, {'token': '6', 'label': 'CARDINAL', 'iob': 3}, {'token': 'Decembe
        {'token': ')', 'label': 'null', 'iob': 2}, {'token': 'is', 'label': 'null', 'iob': 2}, {'token': 'a', 'label':
        {'token': '-', 'label': 'null', 'iob': 2}, {'token': 'Canadian', 'label': 'null', 'iob': 2}, {'token': 'cogniti
        {'token': 'and', 'label': 'null', 'iob': 2}, {'token': 'computer', 'label': 'null', 'iob': 2}, {'token': 'scier
        {'token': 'most', 'label': 'null', 'iob': 2}, {'token': 'noted', 'label': 'null', 'iob': 2}, {'token': 'for', '
        {'token': 'work', 'label': 'null', 'iob': 2}, {'token': 'on', 'label': 'null', 'iob': 2}, {'token': 'artificial
        {'token': 'networks', 'label': 'null', 'iob': 2}, {'token': '.', 'label': 'null', 'iob': 2}
        ... ]}
```

How to build a customized NER service?

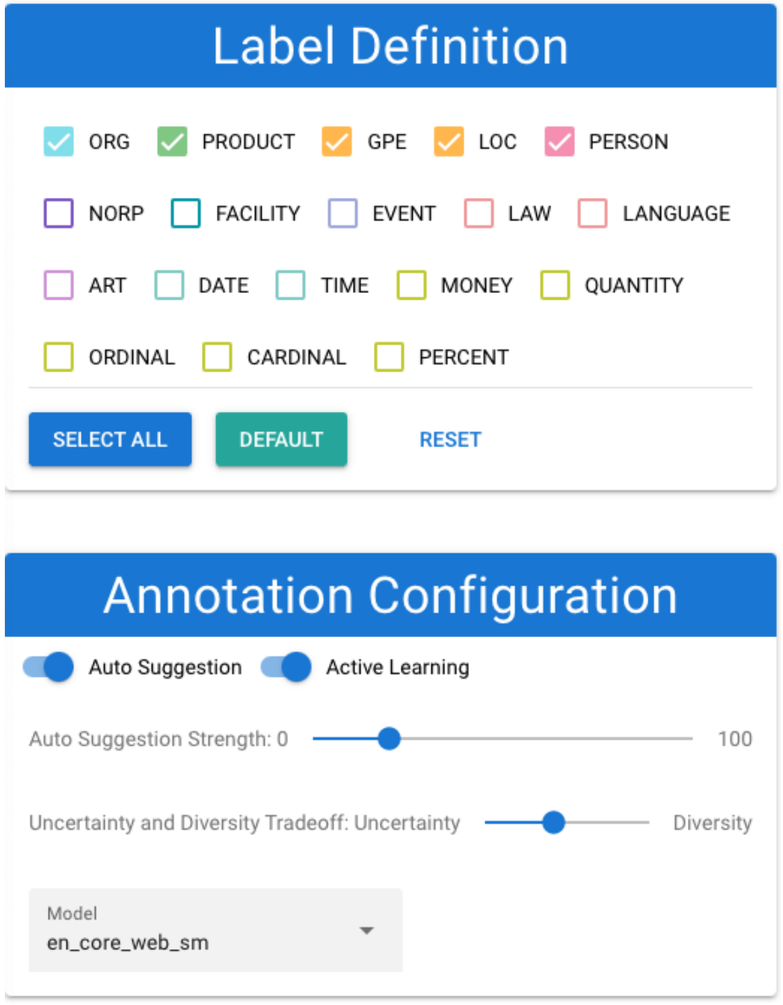
- Data Collection -> Data Annotation -> Model Training & Evaluation -> Model Deployment (API/GUI)

```
>>> raw_dataset[0]
"Geoffrey Everest Hinton (born 6 December 1947) is a British-Canadian cognitive psychologist and computer scientist, mos
```

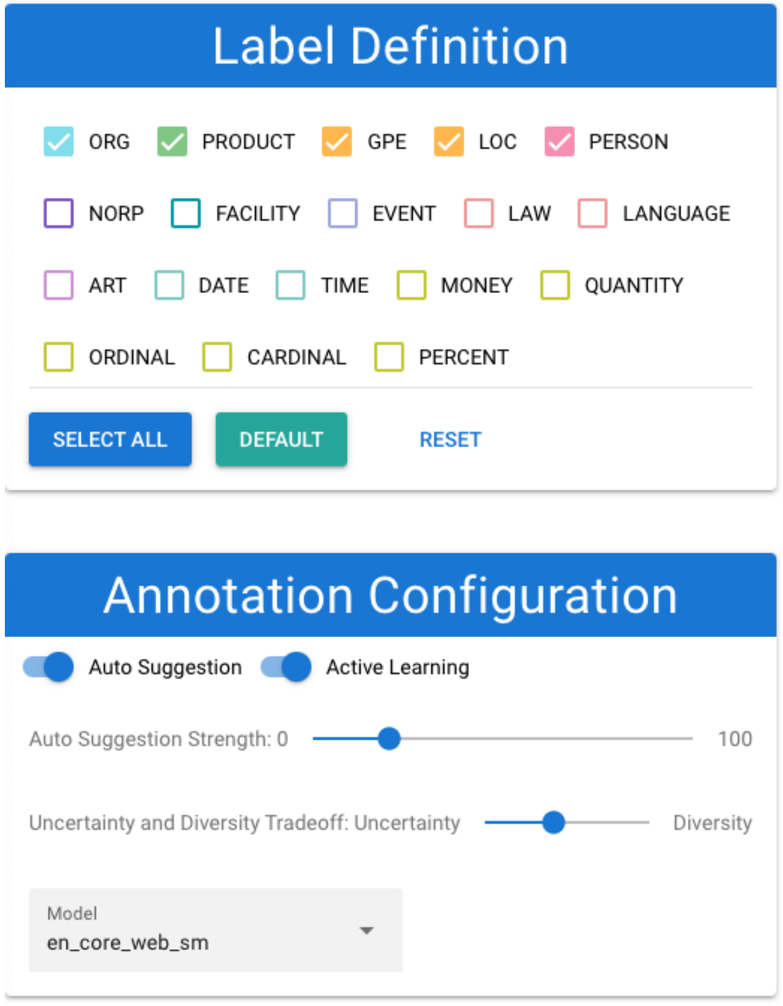
```
>>> train_dataset[0]
{'text': 'Geoffrey Everest Hinton (born 6 December 1947) is a British-Canadian cognitive psychologist and computer scier
'annotation': [{'token': 'Geoffrey', 'label': 'PERSON', 'iob': 3}, {'token': 'Everest', 'label': 'PERSON', 'iob': 1}, {'
        {'token': 'born', 'label': 'null', 'iob': 2}, {'token': '6', 'label': 'CARDINAL', 'iob': 3}, {'token': 'Decembe
        {'token': ')', 'label': 'null', 'iob': 2}, {'token': 'is', 'label': 'null', 'iob': 2}, {'token': 'a', 'label':
        {'token': '-', 'label': 'null', 'iob': 2}, {'token': 'Canadian', 'label': 'null', 'iob': 2}, {'token': 'cogniti
        {'token': 'and', 'label': 'null', 'iob': 2}, {'token': 'computer', 'label': 'null', 'iob': 2}, {'token': 'scier
        {'token': 'most', 'label': 'null', 'iob': 2}, {'token': 'noted', 'label': 'null', 'iob': 2}, {'token': 'for', '
        {'token': 'work', 'label': 'null', 'iob': 2}, {'token': 'on', 'label': 'null', 'iob': 2}, {'token': 'artificial
        {'token': 'networks', 'label': 'null', 'iob': 2}, {'token': '.', 'label': 'null', 'iob': 2}
        ... ]}
```

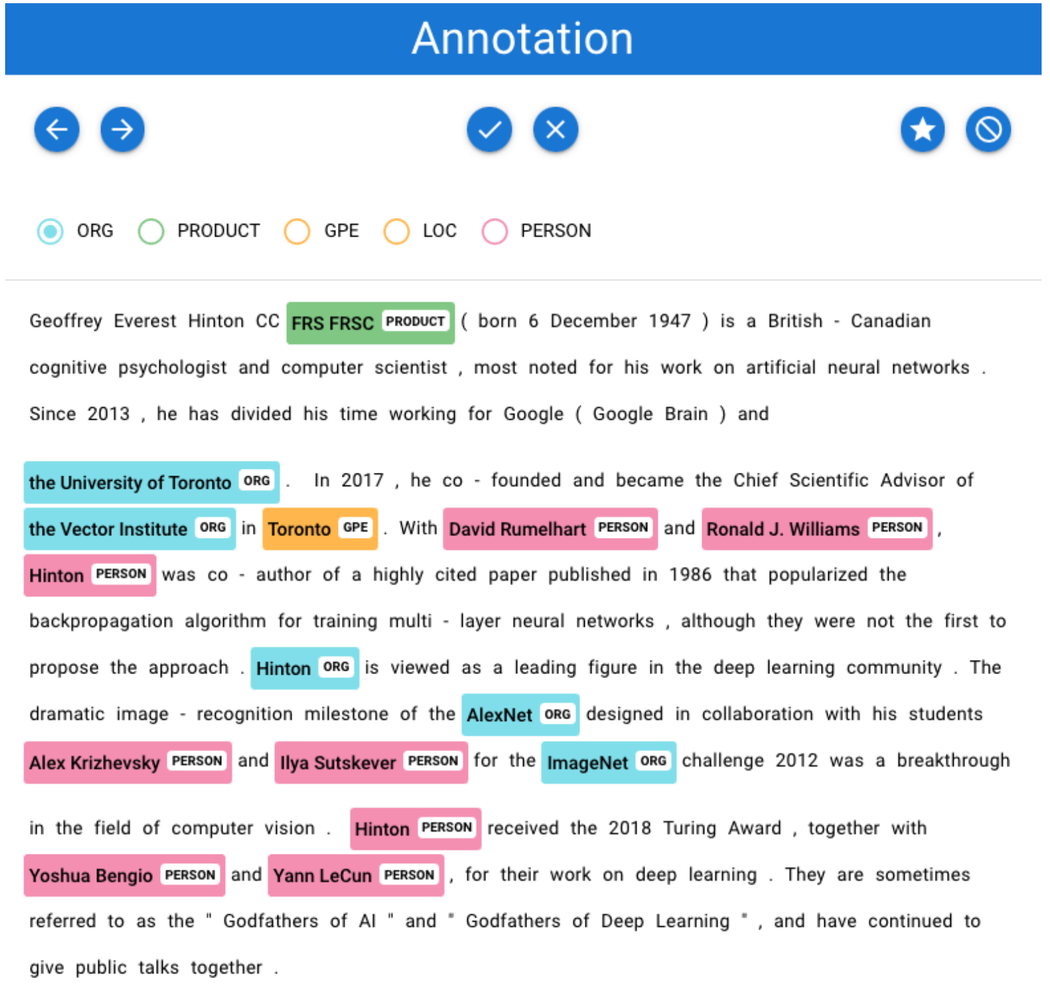
Annotation Interface

Annotation Interface



Annotation Interface





Data Quality Control (In Progress)

```
>>> db.examples[0]
{'text': 'Geoffrey Everest Hinton (born 6 December 1947)
'annotations': [{
  'user1':[{'token': 'Geoffrey', 'label': 'PERSON', 'iob
        {'token': 'Everest', 'label': 'PERSON', 'iob': 1
        {'token': 'Hinton', 'label': 'PERSON', 'iob': 1},
        {'token': 'born', 'label': 'null', 'iob': 2},
        {'token': '6', 'label': 'CARDINAL', 'iob': 3},
        {'token': 'December', 'label': 'DATE', 'iob': 3},
        ... ],
  'user2':[{'token': 'Geoffrey', 'label': 'null', 'iob':
        {'token': 'Everest', 'label': 'null', 'iob': 2},
        {'token': 'Hinton', 'label': 'null', 'iob': 2},
        {'token': 'born', 'label': 'null', 'iob': 2},
        {'token': '6', 'label': 'DATE', 'iob': 3},
        {'token': 'December', 'label': 'DATE', 'iob': 3},
        ... ]}]
```

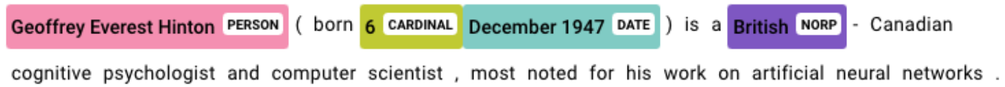
Data Quality Control (In Progress)

```
>>> db.examples[0]
{'text': 'Geoffrey Everest Hinton (born 6 December 1947)
'annotations': [{
  'user1':[{'token': 'Geoffrey', 'label': 'PERSON', 'iob
        {'token': 'Everest', 'label': 'PERSON', 'iob': 1
        {'token': 'Hinton', 'label': 'PERSON', 'iob': 1},
        {'token': 'born', 'label': 'null', 'iob': 2},
        {'token': '6', 'label': 'CARDINAL', 'iob': 3},
        {'token': 'December', 'label': 'DATE', 'iob': 3},
        ... ],
  'user2':[{'token': 'Geoffrey', 'label': 'null', 'iob':
        {'token': 'Everest', 'label': 'null', 'iob': 2},
        {'token': 'Hinton', 'label': 'null', 'iob': 2},
        {'token': 'born', 'label': 'null', 'iob': 2},
        {'token': '6', 'label': 'DATE', 'iob': 3},
        {'token': 'December', 'label': 'DATE', 'iob': 3},
        ... ]}]
```

Data Quality Control (In Progress)

```
>>> db.examples[0]
{'text': 'Geoffrey Everest Hinton (born 6 December 1947)
'annotations': [{
  'user1':[{'token': 'Geoffrey', 'label': 'PERSON', 'iob
        {'token': 'Everest', 'label': 'PERSON', 'iob': 1
        {'token': 'Hinton', 'label': 'PERSON', 'iob': 1},
        {'token': 'born', 'label': 'null', 'iob': 2},
        {'token': '6', 'label': 'CARDINAL', 'iob': 3},
        {'token': 'December', 'label': 'DATE', 'iob': 3},
        ... ],
  'user2':[{'token': 'Geoffrey', 'label': 'null', 'iob':
        {'token': 'Everest', 'label': 'null', 'iob': 2},
        {'token': 'Hinton', 'label': 'null', 'iob': 2},
        {'token': 'born', 'label': 'null', 'iob': 2},
        {'token': '6', 'label': 'DATE', 'iob': 3},
        {'token': 'December', 'label': 'DATE', 'iob': 3},
        ... ]}]
```

- User 1

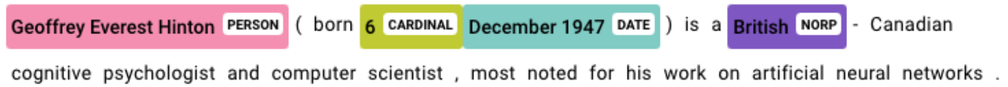


- User 2

Data Quality Control (In Progress)

```
>>> db.examples[0]
{'text': 'Geoffrey Everest Hinton (born 6 December 1947)
'annotations': [{
  'user1':[{'token': 'Geoffrey', 'label': 'PERSON', 'iob
        {'token': 'Everest', 'label': 'PERSON', 'iob': 1
        {'token': 'Hinton', 'label': 'PERSON', 'iob': 1},
        {'token': 'born', 'label': 'null', 'iob': 2},
        {'token': '6', 'label': 'CARDINAL', 'iob': 3},
        {'token': 'December', 'label': 'DATE', 'iob': 3},
        ... ],
  'user2':[{'token': 'Geoffrey', 'label': 'null', 'iob':
        {'token': 'Everest', 'label': 'null', 'iob': 2},
        {'token': 'Hinton', 'label': 'null', 'iob': 2},
        {'token': 'born', 'label': 'null', 'iob': 2},
        {'token': '6', 'label': 'DATE', 'iob': 3},
        {'token': 'December', 'label': 'DATE', 'iob': 3},
        ... ]}]
```

- User 1



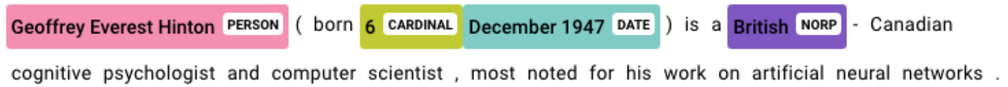
- User 2

- Disagreement Visualization
  - Help needed

Data Quality Control (In Progress)

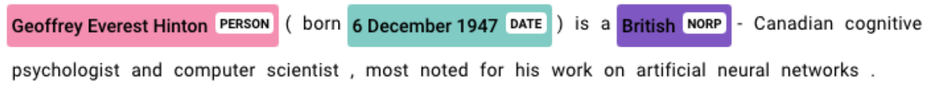
```
>>> db.examples[0]
{'text': 'Geoffrey Everest Hinton (born 6 December 1947)
'annotations': [{
  'user1':[{'token': 'Geoffrey', 'label': 'PERSON', 'iob
        {'token': 'Everest', 'label': 'PERSON', 'iob': 1
        {'token': 'Hinton', 'label': 'PERSON', 'iob': 1},
        {'token': 'born', 'label': 'null', 'iob': 2},
        {'token': '6', 'label': 'CARDINAL', 'iob': 3},
        {'token': 'December', 'label': 'DATE', 'iob': 3},
        ... ],
  'user2':[{'token': 'Geoffrey', 'label': 'null', 'iob':
        {'token': 'Everest', 'label': 'null', 'iob': 2},
        {'token': 'Hinton', 'label': 'null', 'iob': 2},
        {'token': 'born', 'label': 'null', 'iob': 2},
        {'token': '6', 'label': 'DATE', 'iob': 3},
        {'token': 'December', 'label': 'DATE', 'iob': 3},
        ... ]}]
```

- User 1



- User 2

- Disagreement Visualization
  - Help needed
- Golden Standard



Model Training

```
from transformers import DistilBertForTokenClassification, Trainer, TrainingArguments
training_args = TrainingArguments(
    num_train_epochs=3,              # total number of training epochs
    per_device_train_batch_size=16,  # batch size per device during training
    per_device_eval_batch_size=64,   # batch size for evaluation
    warmup_steps=500,                # number of warmup steps for learning rate scheduler
    weight_decay=0.01,               # strength of weight decay
)

model = DistilBertForTokenClassification.from_pretrained('distilbert-base-cased', num_labels=len(unique_tags))

trainer = Trainer(
    model=model,                         # the instantiated 🤗 Transformers model to be trained
    args=training_args,                  # training arguments, defined above
    train_dataset=train_dataset,         # training dataset
    eval_dataset=val_dataset             # evaluation dataset
)

trainer.train()
```

Model Training

```
from transformers import DistilBertForTokenClassification, Trainer, TrainingArguments
training_args = TrainingArguments(
    num_train_epochs=3,              # total number of training epochs
    per_device_train_batch_size=16,  # batch size per device during training
    per_device_eval_batch_size=64,   # batch size for evaluation
    warmup_steps=500,                # number of warmup steps for learning rate scheduler
    weight_decay=0.01,               # strength of weight decay
)

model = DistilBertForTokenClassification.from_pretrained('distilbert-base-cased', num_labels=len(unique_tags))

trainer = Trainer(
    model=model,                         # the instantiated 🤗 Transformers model to be trained
    args=training_args,                  # training arguments, defined above
    train_dataset=train_dataset,         # training dataset
    eval_dataset=val_dataset             # evaluation dataset
)

trainer.train()
```

Model Training

```
from transformers import DistilBertForTokenClassification, Trainer, TrainingArguments
training_args = TrainingArguments(
    num_train_epochs=3,              # total number of training epochs
    per_device_train_batch_size=16,  # batch size per device during training
    per_device_eval_batch_size=64,   # batch size for evaluation
    warmup_steps=500,                # number of warmup steps for learning rate scheduler
    weight_decay=0.01,               # strength of weight decay
)

model = DistilBertForTokenClassification.from_pretrained('distilbert-base-cased', num_labels=len(unique_tags))

trainer = Trainer(
    model=model,                         # the instantiated 🤗 Transformers model to be trained
    args=training_args,                  # training arguments, defined above
    train_dataset=train_dataset,         # training dataset
    eval_dataset=val_dataset             # evaluation dataset
)

trainer.train()
```

Model Training

```
from transformers import DistilBertForTokenClassification, Trainer, TrainingArguments
training_args = TrainingArguments(
    num_train_epochs=3,              # total number of training epochs
    per_device_train_batch_size=16,  # batch size per device during training
    per_device_eval_batch_size=64,   # batch size for evaluation
    warmup_steps=500,                # number of warmup steps for learning rate scheduler
    weight_decay=0.01,               # strength of weight decay
)

model = DistilBertForTokenClassification.from_pretrained('distilbert-base-cased', num_labels=len(unique_tags))

trainer = Trainer(
    model=model,                         # the instantiated 🤗 Transformers model to be trained
    args=training_args,                  # training arguments, defined above
    train_dataset=train_dataset,         # training dataset
    eval_dataset=val_dataset             # evaluation dataset
)

trainer.train()
```

Model Training

```
from transformers import DistilBertForTokenClassification, Trainer, TrainingArguments
training_args = TrainingArguments(
    num_train_epochs=3,              # total number of training epochs
    per_device_train_batch_size=16,  # batch size per device during training
    per_device_eval_batch_size=64,   # batch size for evaluation
    warmup_steps=500,                # number of warmup steps for learning rate scheduler
    weight_decay=0.01,               # strength of weight decay
)

model = DistilBertForTokenClassification.from_pretrained('distilbert-base-cased', num_labels=len(unique_tags))

trainer = Trainer(
    model=model,                         # the instantiated 🤗 Transformers model to be trained
    args=training_args,                  # training arguments, defined above
    train_dataset=train_dataset,         # training dataset
    eval_dataset=val_dataset             # evaluation dataset
)

trainer.train()
```

Model Training

```
from transformers import DistilBertForTokenClassification, Trainer, TrainingArguments
training_args = TrainingArguments(
    num_train_epochs=3,              # total number of training epochs
    per_device_train_batch_size=16,  # batch size per device during training
    per_device_eval_batch_size=64,   # batch size for evaluation
    warmup_steps=500,                # number of warmup steps for learning rate scheduler
    weight_decay=0.01,               # strength of weight decay
)

model = DistilBertForTokenClassification.from_pretrained('distilbert-base-cased', num_labels=len(unique_tags))

trainer = Trainer(
    model=model,                         # the instantiated 🤗 Transformers model to be trained
    args=training_args,                  # training arguments, defined above
    train_dataset=train_dataset,         # training dataset
    eval_dataset=val_dataset             # evaluation dataset
)

trainer.train()
```

Model Training

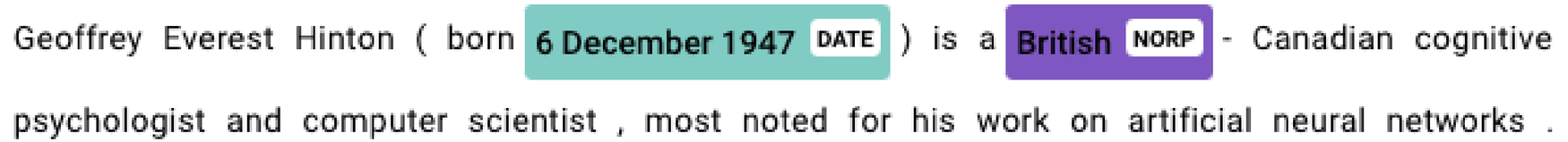
```
from transformers import DistilBertForTokenClassification, Trainer, TrainingArguments
training_args = TrainingArguments(
    num_train_epochs=3,              # total number of training epochs
    per_device_train_batch_size=16,  # batch size per device during training
    per_device_eval_batch_size=64,   # batch size for evaluation
    warmup_steps=500,                # number of warmup steps for learning rate scheduler
    weight_decay=0.01,               # strength of weight decay
)

model = DistilBertForTokenClassification.from_pretrained('distilbert-base-cased', num_labels=len(unique_tags))

trainer = Trainer(
    model=model,                         # the instantiated 🤗 Transformers model to be trained
    args=training_args,                  # training arguments, defined above
    train_dataset=train_dataset,         # training dataset
    eval_dataset=val_dataset             # evaluation dataset
)

trainer.train()
```

Model Selection using A/B Test (In progress)

Model Selection using A/B Test (In progress)

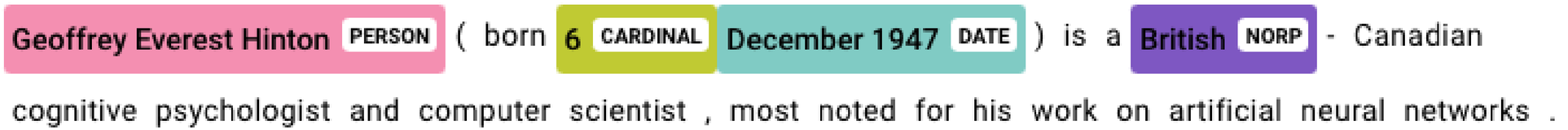
- Prediction from Trained Model 1

- Prediction from Trained Model 2

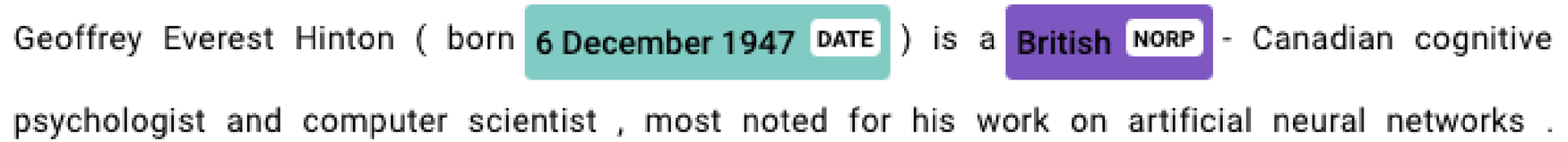


Model Selection using A/B Test (In progress)

- Prediction from Trained Model 1



- Prediction from Trained Model 2



- They both make one mistake and have the same accuracy. Which model to choose?
  - Let user make the decision.
  - E.g. A/B Test with 100 examples.
    - Model 1 has 68/100 "likes" and Model 2 has 55/100 "likes" -> Deploy model 1

Next Steps

- Functionality
  - Complete the data quality control interface
  - Model Selection with A/B testing
  - Data annotation with active learning
- Experiement
  - Benchmark the performance of different active learning algorithm on NER task
- Demo
  - Build a customized NER service for a specific domain (Medical Data)