

Textual Data Analysis

Classification and Explanation of Predictions



TURKUNLP
.ORG



**UNIVERSITY
OF TURKU**

/ Text classification

- **Input:** text, represented as sequence of tokens
- **Output:**
 - One label/class for the whole text (multi-class) or
 - Several labels/classes for the whole text (multi-label)

Note: This material should be mostly familiar from the Deep Learning in HLT course, where we went through classification in some detail, so I will recap only very briefly - enough for you to hang on, though



/ Text classification

Text classification
Document classification
Sequence classification

positive



This is a good movie.

The	DET
dog	NOUN
runs	VERB
in	ADP
the	DET
park	NOUN
.	PUNCT

Sequence labeling
Sequence tagging
Token classification

Don't copy, modify, resell, distribute or reverse engineer this app.



Niantic grants you a limited nonexclusive nontransferable non sublicensable license to download and install a copy of the app on a mobile device and to run such copy of the app solely for your own personal noncommercial purposes. Except as expressly permitted in these terms you may not a) copy, modify or create derivative works based on the app, b) distribute transfer sublicense lease lend or rent the app to any third party, c) reverse engineer decompile or disassemble the app...

(Example from [Manor and Li, 2019.](#))

Sequence to sequence
Text generation

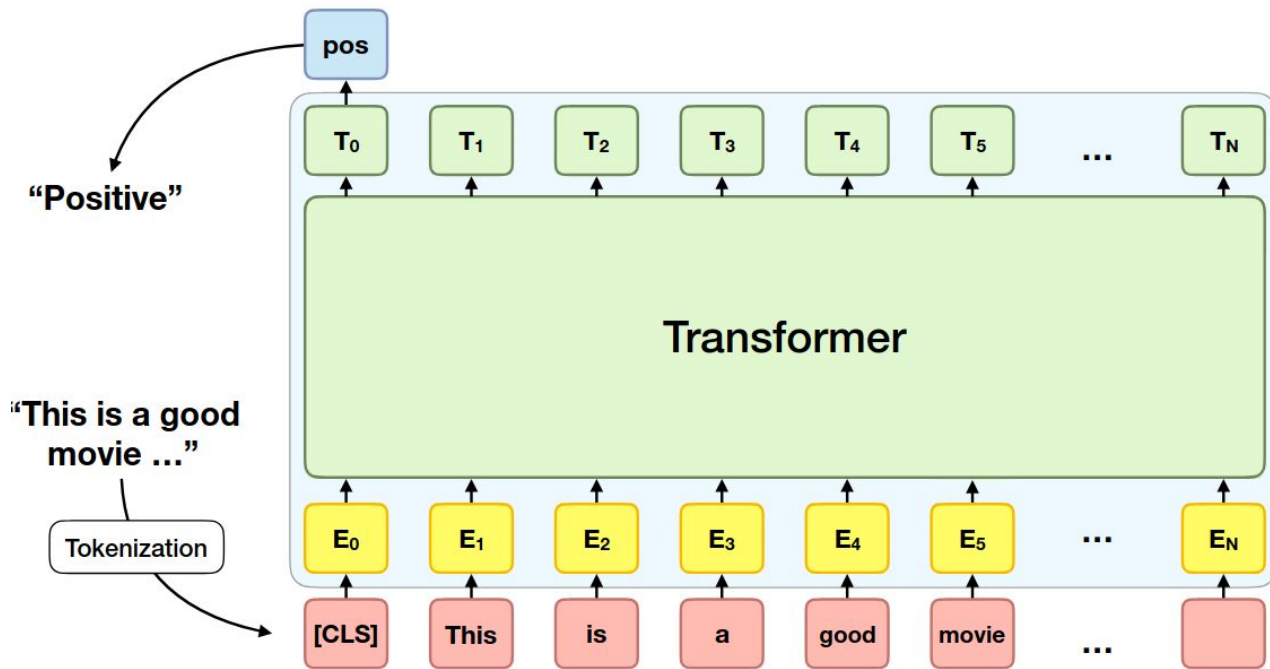


TURKUNLP
.ORG



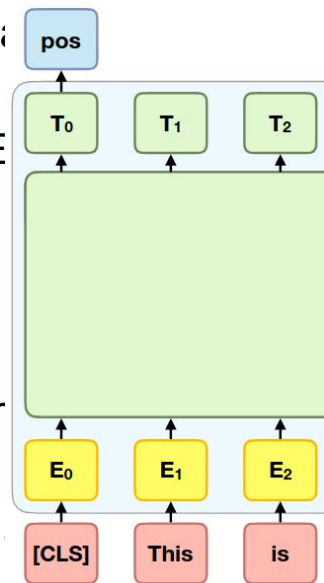
**UNIVERSITY
OF TURKU**

/ Recap: Classification with transformers

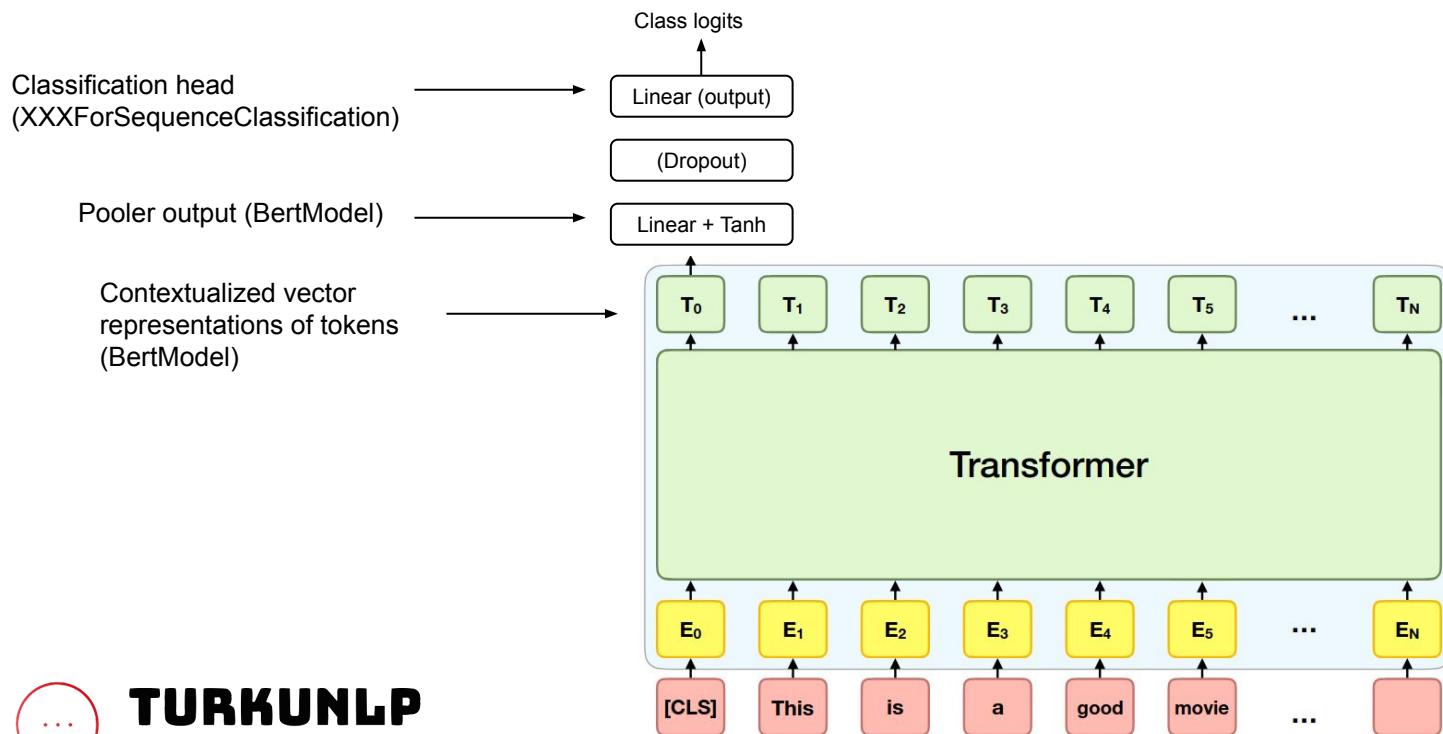


/ Recap: Classification with transformers

- BERT and many related models: classification head attached to special token added to start of token sequence
- Many models (e.g. XLM-R) do not have the [CLS] token trained as BERT [CLS], but may have a special first token nonetheless
 - Classification based on this 1st token
 - Or a pool across output layer (e.g. mean/max/...)
- Additional, randomly initialized output layer added to the pre-trained model
- Fine-tuning can either only train this layer (faster) or continue training other parts of the model



/ Recap: Classification with transformers



```

class XLRobertaClassificationHead(nn.Module):
    """Head for sentence-level classification tasks."""

    def __init__(self, config):
        super().__init__()
        self.dense = nn.Linear(config.hidden_size, config.hidden_size)
        classifier_dropout = (
            config.classifier_dropout if config.classifier_dropout is not None else config
        )
        self.dropout = nn.Dropout(classifier_dropout)
        self.out_proj = nn.Linear(config.hidden_size, config.num_labels)

    def forward(self, features, **kwargs):
        x = features[:, 0, :] # take <s> token (equiv. to [CLS])
        x = self.dropout(x)
        x = self.dense(x)
        x = torch.tanh(x)
        x = self.dropout(x)
        x = self.out_proj(x)
        return x

```

Hidden layer (of the head)

Output logits

```
tokenizer=transformers.AutoTokenizer.from_pretrained("FacebookAI/xlm-roberta-base")
tok=tokenizer("Hello there! This is an example text.", "And a second sequence.")
print(tok)
pprint.pprint(tokenizer.convert_ids_to_tokens(tok["input_ids"]),compact=True)
```

```
{'input_ids': [0, 35378, 2685, 38, 3293, 83, 142, 27781, 7986, 5, 2, 2, 3493, 10, 1]
['<s>', 'Hello', 'there', '!', 'This', 'is', 'an', 'example', 'text',
'</s>', '</s>', 'And', 'a', 'second', 'se', 'que', 'nce', '.'],
['</s>']
```

```
tokenizer=transformers.AutoTokenizer.from_pretrained("google-bert/bert-base-cased")
tok=tokenizer("Hello there! This is an example text.", "And a second sequence.")
print(tok)
pprint.pprint(tokenizer.convert_ids_to_tokens(tok["input_ids"]),compact=True)
```

```
{'input_ids': [101, 8667, 1175, 106, 1188, 1110, 1126, 1859, 3087, 119, 102, 1262, 101]
['[CLS]', 'Hello', 'there', '!', 'This', 'is', 'an', 'example', 'text', '.',
'[SEP]', 'And', 'a', 'second', 'sequence', '.', '[SEP]']
```


/ Alternative approaches

- So far: pre-trained model, fine-tuned on annotated data
- Except for the “pre-trained” bit, this is the standard ML paradigm:
 - Pick a task
 - Design the label set and annotation guidelines
 - Make enough data by hand...
 - ... to train a model to fit the data well-enough
 - Apply on new data



/ Text classification traditional modes

Approach	Rules by	Applied by	Applied to	Quality	Speed
Hand-coding	<div>Human</div> <div><i>conceptual rules</i> <i>meaning-oriented application</i></div> <div>Human</div>	100% of texts	<div>High</div> <div>Limited</div> <div>High</div> <div>Both improve with information density of language representation, see: Figure 2.</div>	<div>Slow</div> <div>Fast</div> <div>Fast</div>	
Dictionary	<div>Human</div> <div><i>machine-readable rules</i> <i>mechanical application</i></div> <div>Machine</div>	100% of texts			
Supervised Machine Learning	<div>Human</div> <div><i>= hand-coding</i></div> <div>Human</div>	Fraction			
	<div>Machine</div> <div><i>mechanical application</i></div> <div>Machine</div>	Majority			
<div>train model = extract machine-readable rules from human coding</div>				of texts	



/ Zero/few-shot

Approach	Rules by	Applied by	Applied to	Quality	Speed
Hand-coding	<div>Human</div> <div><i>conceptual rules</i> <i>meaning-oriented application</i></div>	<div>Human</div>	100% of texts	<div>High</div>	Slow



Approach	Rules by	Applied by	Applied to	Quality	Speed
Hand-coding	<div>Human</div> <div><i>conceptual rules</i> <i>meaning-oriented application</i></div>	<div>LLM</div>	100% of texts	<div>?</div>	<div>?</div>



/Zero/few-shot

- Zero-shot: describe the task and labels, give an instance, let an LLM assign the label
- Few-shot: as before, but also give several examples of labeled instances
- Advantage:
 - Data annotation step skipped - no need to invest the effort
- Disadvantage:
 - Data annotation step skipped - it is important to have your task conception face the real data
 - Powerful LLMs are not cheap, nor fast to run



/ Zero/few shot -> does it work?

- “It depends” as usual
- But let us go through one recent, somewhat careful study:
<https://arxiv.org/pdf/2406.08660v1>
- Explicit comparison of zero-shot against fine-tuned encoder models on 4 tasks



/ Prompt 1/4

Sentiment Analysis on The New York Times Coverage of the US Economy

Prompt:

You have been assigned the task of zero-shot text classification for sentiment analysis. Your objective is to classify a given text snippet into one of several possible class labels, based on the sentiment expressed in the text. Your output should consist of a single class label that best matches the sentiment expressed in the text. Your output should consist of a single class label that best matches the given text. Choose ONLY from the given class labels below and ONLY output the label without any other characters.

Text: <Text>

Labels: 'Negative Sentiment', 'Positive Sentiment'

Answer:

/ Prompt 2/4

Stance Classification on Tweets about Kavanaugh Nomination

Prompt:

You have been assigned the task of zero-shot text classification for stance classification. Your objective is to classify a given text snippet into one of several possible class labels, based on the attitudinal stance towards the given text. Your output should consist of a single class label that best matches the stance expressed in the text. Your output should consist of a single class label that best matches the given text. Choose ONLY from the given class labels below and ONLY output the label without any other characters.

Text: <Text>

Labels: 'negative attitudinal stance towards', 'positive attitudinal stance towards'

Answer:



/ Prompt 3/4

Emotion Detection on Political Texts in German

Prompt:

You have been assigned the task of zero-shot text classification for emotion classification. Your objective is to classify a given text snippet into one of several possible class labels, based on the anger level in the given text. Your output should consist of a single class label that best matches the anger expressed in the text. Choose **ONLY** from the given class labels below and **ONLY** output the label without any other characters.

Text: <Text>

Labels: 'Angry', 'Non-Angry'

Answer:



/ Prompt 4/4

Multi-Class Stance Classification on Parties' EU Positions

Prompt:

You have been assigned the task of zero-shot text classification for political texts on attitudinal stance towards Brexit and leave demands related to the European Union (EU). Your objective is to classify a given text snippet into one of several possible class labels, based on the stance towards Brexit and general leave demands in the given text. Your output should consist of a single class label that best matches the content expressed in the text. Choose ONLY from the given class labels below and ONLY output the label without any other characters.

Text: <Text>

Labels: 'Neutral towards Leave demands', 'Pro-Leave demands', 'Very Pro-Leave demands'

Answer:



/ Task 1/4

Table 1: Results for Sentiment Analysis (US Economy)

Model Name	Accuracy	Prec. (wgt.)	Recall (wgt.)	F1 (macro)	F1 (wgt.)
MAJ-VOT	0.73 (± 0.00)	0.53 (± 0.00)	0.73 (± 0.00)	0.42 (± 0.00)	0.61 (± 0.00)
ROB-BASE	0.89 (± 0.00)	0.89 (± 0.01)	0.89 (± 0.00)	0.86 (± 0.01)	0.89 (± 0.01)
ROB-LRG	0.92 (± 0.01)	0.92 (± 0.01)	0.92 (± 0.01)	0.90 (± 0.01)	0.92 (± 0.01)
DEB-V3	0.92 (± 0.02)	0.92 (± 0.01)	0.92 (± 0.02)	0.90 (± 0.02)	0.92 (± 0.01)
ELE-LRG	0.90 (± 0.01)	0.90 (± 0.01)	0.90 (± 0.01)	0.88 (± 0.02)	0.90 (± 0.01)
XLNET-LRG	0.81 (± 0.01)	0.85 (± 0.01)	0.81 (± 0.01)	0.78 (± 0.01)	0.82 (± 0.01)
BART-LRG	0.85 (± 0.00)	0.84 (± 0.00)	0.85 (± 0.00)	0.80 (± 0.00)	0.84 (± 0.00)
GPT-3.5	0.82 (± 0.00)	0.84 (± 0.00)	0.82 (± 0.00)	0.79 (± 0.00)	0.83 (± 0.00)
GPT-4	0.87 (± 0.00)	0.87 (± 0.00)	0.87 (± 0.00)	0.84 (± 0.00)	0.87 (± 0.00)
CLD-OPUS	0.86 (± 0.00)	0.87 (± 0.00)	0.86 (± 0.00)	0.83 (± 0.00)	0.87 (± 0.00)

/ Task 2/4

Table 2: Results for Stance Classification (Nomination Approval)

Model Name	Accuracy	Prec. (wgt.)	Recall (wgt.)	F1 (macro)	F1 (wgt.)
MAJ-VOT	0.50 (± 0.00)	0.25 (± 0.00)	0.50 (± 0.00)	0.33 (± 0.00)	0.33 (± 0.00)
ROB-BASE	0.86 (± 0.01)	0.86 (± 0.01)	0.86 (± 0.01)	0.86 (± 0.01)	0.86 (± 0.01)
ROB-LRG	0.92 (± 0.01)	0.93 (± 0.01)	0.92 (± 0.01)	0.92 (± 0.01)	0.92 (± 0.01)
DEB-V3	0.94 (± 0.01)	0.94 (± 0.01)	0.94 (± 0.01)	0.93 (± 0.01)	0.94 (± 0.01)
ELE-LRG	0.74 (± 0.01)	0.66 (± 0.02)	0.74 (± 0.01)	0.67 (± 0.02)	0.69 (± 0.02)
XLNET-LRG	0.83 (± 0.01)	0.83 (± 0.01)	0.83 (± 0.01)	0.83 (± 0.01)	0.83 (± 0.01)
BART-LRG	0.53 (± 0.00)	0.59 (± 0.00)	0.53 (± 0.00)	0.44 (± 0.00)	0.44 (± 0.00)
GPT-3.5	0.53 (± 0.00)	0.58 (± 0.00)	0.53 (± 0.00)	0.48 (± 0.00)	0.47 (± 0.00)
GPT-4	0.58 (± 0.00)	0.68 (± 0.00)	0.58 (± 0.00)	0.51 (± 0.00)	0.51 (± 0.00)
CLD-OPUS	0.61 (± 0.00)	0.68 (± 0.00)	0.61 (± 0.00)	0.57 (± 0.00)	0.57 (± 0.00)

/ Task 3/4

Table 3: Results for Emotion Detection (Anger)

Model Name	Accuracy	Prec. (wgt.)	Recall (wgt.)	F1 (macro)	F1 (wgt.)
MAJ-VOT	0.71 (± 0.00)	0.51 (± 0.00)	0.71 (± 0.00)	0.42 (± 0.00)	0.59 (± 0.00)
ROB-BASE	0.87 (± 0.01)	0.88 (± 0.01)	0.87 (± 0.01)	0.82 (± 0.01)	0.88 (± 0.01)
ROB-LRG	0.88 (± 0.01)	0.88 (± 0.00)	0.88 (± 0.01)	0.83 (± 0.00)	0.88 (± 0.00)
DEB-V3	0.88 (± 0.01)	0.88 (± 0.00)	0.88 (± 0.01)	0.83 (± 0.01)	0.88 (± 0.00)
ELE-LRG	0.88 (± 0.00)	0.88 (± 0.02)	0.88 (± 0.00)	0.84 (± 0.00)	0.88 (± 0.00)
XLNET-LRG	0.89 (± 0.00)	0.89 (± 0.00)	0.89 (± 0.00)	0.85 (± 0.00)	0.89 (± 0.00)
ELE-BS-GER	0.88 (± 0.01)	0.88 (± 0.01)	0.88 (± 0.01)	0.83 (± 0.02)	0.88 (± 0.01)
BART-LRG	0.26 (± 0.00)	0.36 (± 0.00)	0.26 (± 0.00)	0.24 (± 0.00)	0.29 (± 0.00)
GPT-3.5	0.15 (± 0.00)	0.23 (± 0.00)	0.15 (± 0.00)	0.15 (± 0.00)	0.16 (± 0.00)
GPT-4	0.20 (± 0.00)	0.18 (± 0.00)	0.20 (± 0.00)	0.18 (± 0.00)	0.13 (± 0.00)
CLD-OPUS	0.15 (± 0.00)	0.16 (± 0.00)	0.15 (± 0.00)	0.14 (± 0.00)	0.11 (± 0.00)

/ Task 4/4

Table 4: Results for Multi-Class Stance Classification (EU Positions)

Model Name	Accuracy	Prec. (wgt.)	Recall (wgt.)	F1 (macro)	F1 (wgt.)
MAJ-VOT	0.83 (± 0.00)	0.68 (± 0.00)	0.83 (± 0.00)	0.30 (± 0.00)	0.75 (± 0.00)
ROB-BASE	0.84 (± 0.00)	0.87 (± 0.01)	0.84 (± 0.00)	0.70 (± 0.02)	0.85 (± 0.00)
ROB-LRG	0.88 (± 0.01)	0.88 (± 0.01)	0.88 (± 0.01)	0.72 (± 0.03)	0.87 (± 0.01)
DEB-V3	0.92 (± 0.01)	0.91 (± 0.01)	0.92 (± 0.01)	0.82 (± 0.02)	0.91 (± 0.01)
ELE-LRG	0.88 (± 0.01)	0.88 (± 0.01)	0.88 (± 0.01)	0.75 (± 0.03)	0.87 (± 0.01)
XLNET-LRG	0.87 (± 0.01)	0.89 (± 0.01)	0.87 (± 0.01)	0.75 (± 0.02)	0.88 (± 0.01)
BART-LRG	0.82 (± 0.00)	0.77 (± 0.00)	0.82 (± 0.00)	0.34 (± 0.00)	0.75 (± 0.00)
GPT-3.5	0.24 (± 0.00)	0.65 (± 0.00)	0.24 (± 0.00)	0.17 (± 0.00)	0.27 (± 0.00)
GPT-4	0.38 (± 0.00)	0.73 (± 0.00)	0.38 (± 0.00)	0.26 (± 0.00)	0.45 (± 0.00)
CLD-OPUS	0.26 (± 0.00)	0.75 (± 0.00)	0.26 (± 0.00)	0.25 (± 0.00)	0.29 (± 0.00)

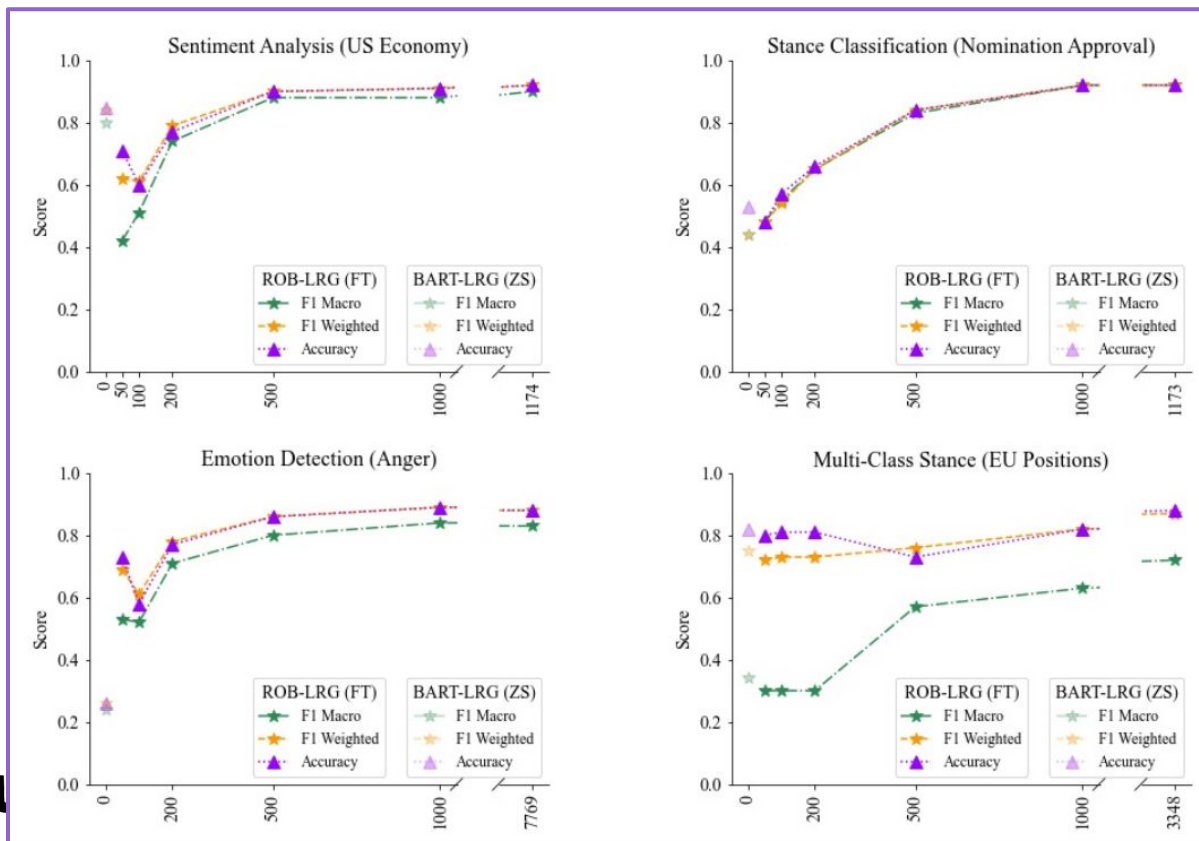


/ Zero/few-shot - did it work?

- In this particular evaluation, on these particular datasets, LLMs underperformed the prior generation of encoder-based classifiers
- That does not always need to be the case
- But it is still the commonsense understanding in the field:
AI hype does not automatically translate to good numbers!
- But how about the annotation effort:



Effect of training set size

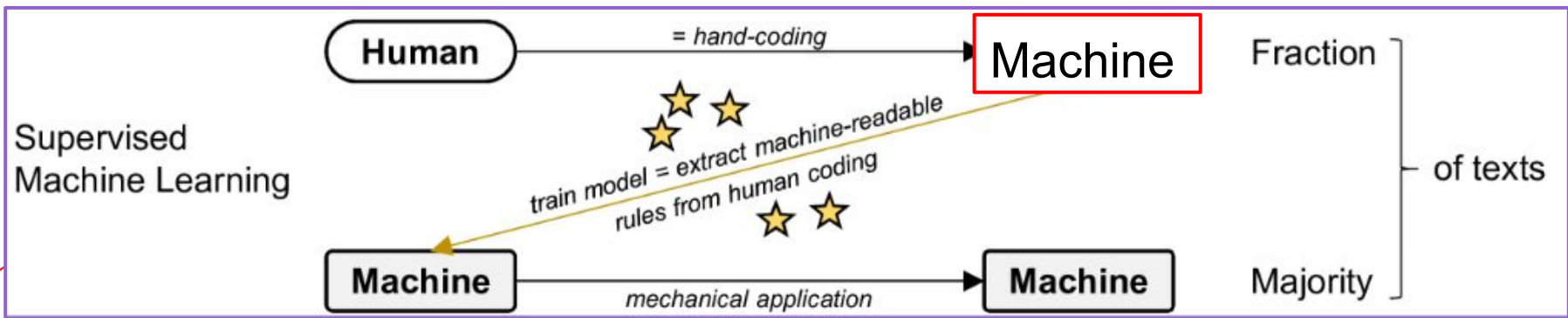
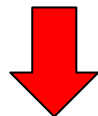
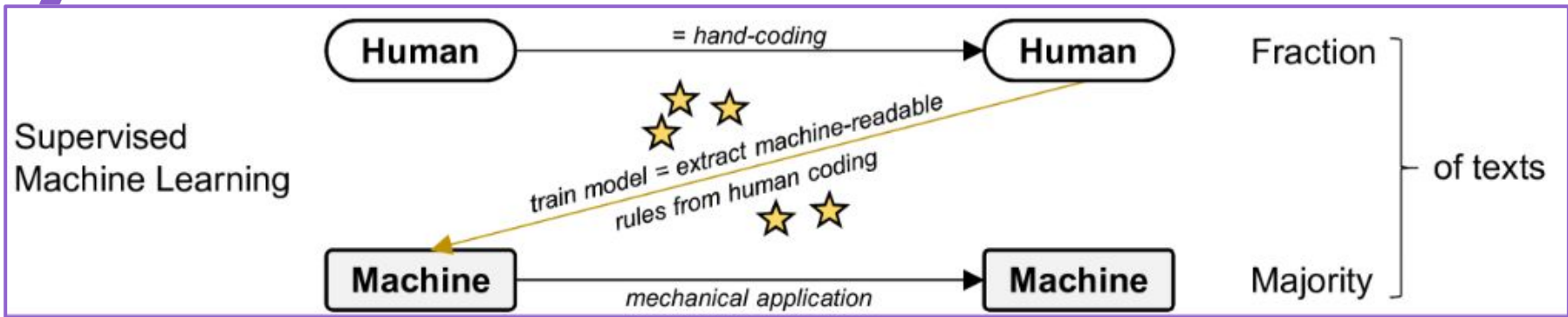


/ Effect of training set size

- In the 4 reported datasets, near-final performance reached after few thousand examples
- That is not very many...
- Conclusion in these particular cases:
 - With small-to-moderate annotation effort...
 - ...and encoder-only models that are comparatively fast and cheap...
 - ...you notably outperform zero-shot strong LLMs
 - Especially so in less common tasks



LLM as annotator



/ LLM as annotator

- A hybrid approach
- Use LLM to annotate training data for an encoder
- Apply the encoder at scale
- Advantages:
 - Leaves out the manual annotation effort
 - Encoder much faster and cheaper to apply at scale
- Disadvantages:
 - The task needs to be simple enough for the LLM
 - Also when recall not critical (can set prediction score cutoff high)



/ LLM as annotator - data cleanup

- Data cleaning in large crawled corpora is an emerging application of this approach
- The task seems well suited for LLMs but it is impossible to apply an LLM on 15T token scale

To ensure Llama 3 is trained on data of the highest quality, we developed a series of data-filtering pipelines. These pipelines include using heuristic filters, NSFW filters, semantic deduplication approaches, and text classifiers to predict data quality. **We found that previous generations of Llama are surprisingly good at identifying high-quality data, hence we used Llama 2 to generate the training data for the text-quality classifiers that are powering Llama 3.**



/ LLM as annotator - data cleanup

- FineWeb-EDU (you had an exercise on this, so this should be familiar to you)
- Llama-3-70B scored 460,000 randomly sampled webpages
- Ranked on a scale of 0-5 for educational content - additive prompt
- Trained an approximately BERT-base sized model as a classifier (numerical regressor in this particular case)
- Applied on the whole of FineWeb

We use the following prompt template to generate document annotations using the Llama3 model:

Below is an extract from a web page. Evaluate whether the page has a high educational value and could be useful in an educational setting for teaching from primary school to grade school levels using the additive 5-point scoring system described below. Points are accumulated based on the satisfaction of each criterion:

- Add 1 point if the extract provides some basic information relevant to educational topics, even if it includes some irrelevant or non-academic content like advertisements and promotional material.
- Add another point if the extract addresses certain elements pertinent to education but does not align closely with educational standards. It might mix educational content with non-educational material, offering a superficial overview of potentially useful topics, or presenting information in a disorganized manner and incoherent writing style.
- Award a third point if the extract is appropriate for educational use and introduces key concepts relevant to school curricula. It is coherent though it may not be comprehensive or could include some extraneous information. It may resemble an introductory section of a textbook or a basic tutorial that is suitable for learning but has notable limitations like treating concepts that are too complex for grade school students.
- Grant a fourth point if the extract is highly relevant and beneficial for educational purposes for a level not higher than grade school, exhibiting a clear and consistent writing style. It could be similar to a chapter from a textbook or a tutorial, offering substantial educational content, including exercises and solutions, with minimal irrelevant information, and the concepts aren't too advanced for grade school students. The content is coherent, focused, and valuable for structured learning.
- Bestow a fifth point if the extract is outstanding in its educational value, perfectly suited for teaching either at primary school or grade school. It follows detailed reasoning, the writing style is easy to follow and offers profound and thorough insights into the subject matter, devoid of any non-educational or complex content.

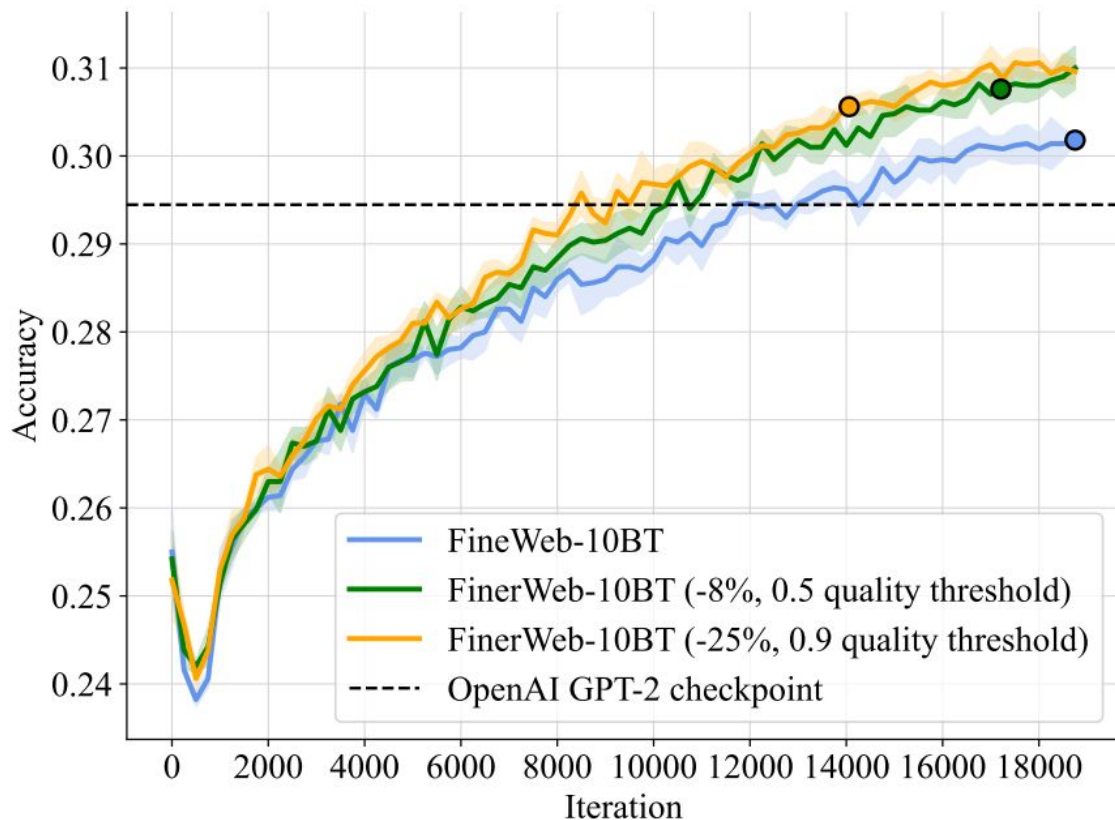
The extract: <EXAMPLE>.

After examining the extract:

- Briefly justify your total score, up to 100 words.
- Conclude with the score using the format: "Educational score: <total points>"

LLM as annotator - line-level data cleanup

- Our own work
- Line-level FineWeb cleanup
- GPT4 + small model
- Further improvement of the data



/ Explaining the predictions

- In many applications, simple prediction is not enough
- Ideally, one would *explain and justify* the prediction
- Encoders with classification heads simply predict per-class logits, not explanations
- LLMs can (with differing success and reliability) justify own predictions
- In the following, I will borrow some slides from this excellent NeurIPS tutorial on model explainability (hereby acknowledging the source!)

<https://explainml-tutorial.github.io/neurips20>

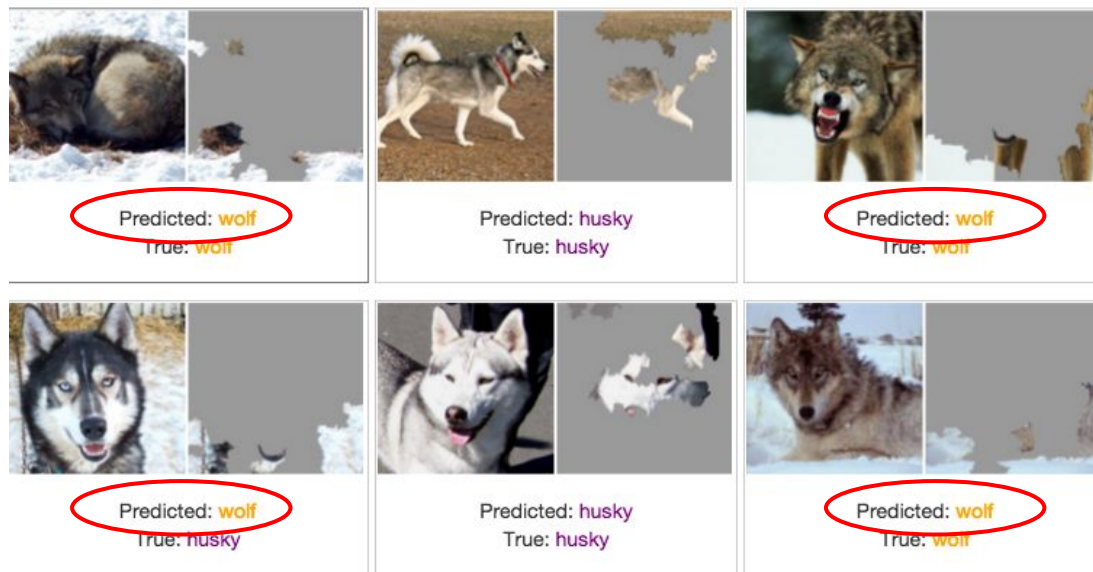


Predict Wolf vs Husky

Only 1 mistake!

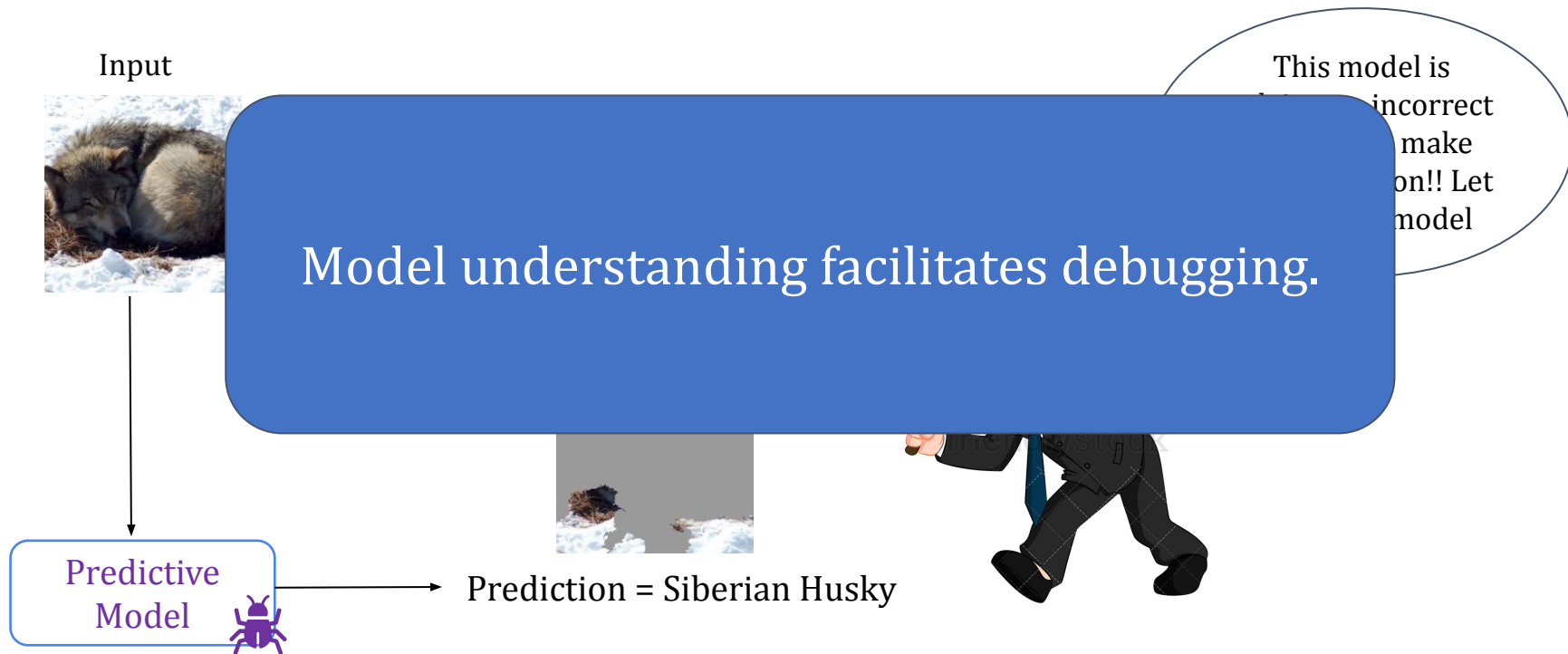


Predict Wolf vs Husky

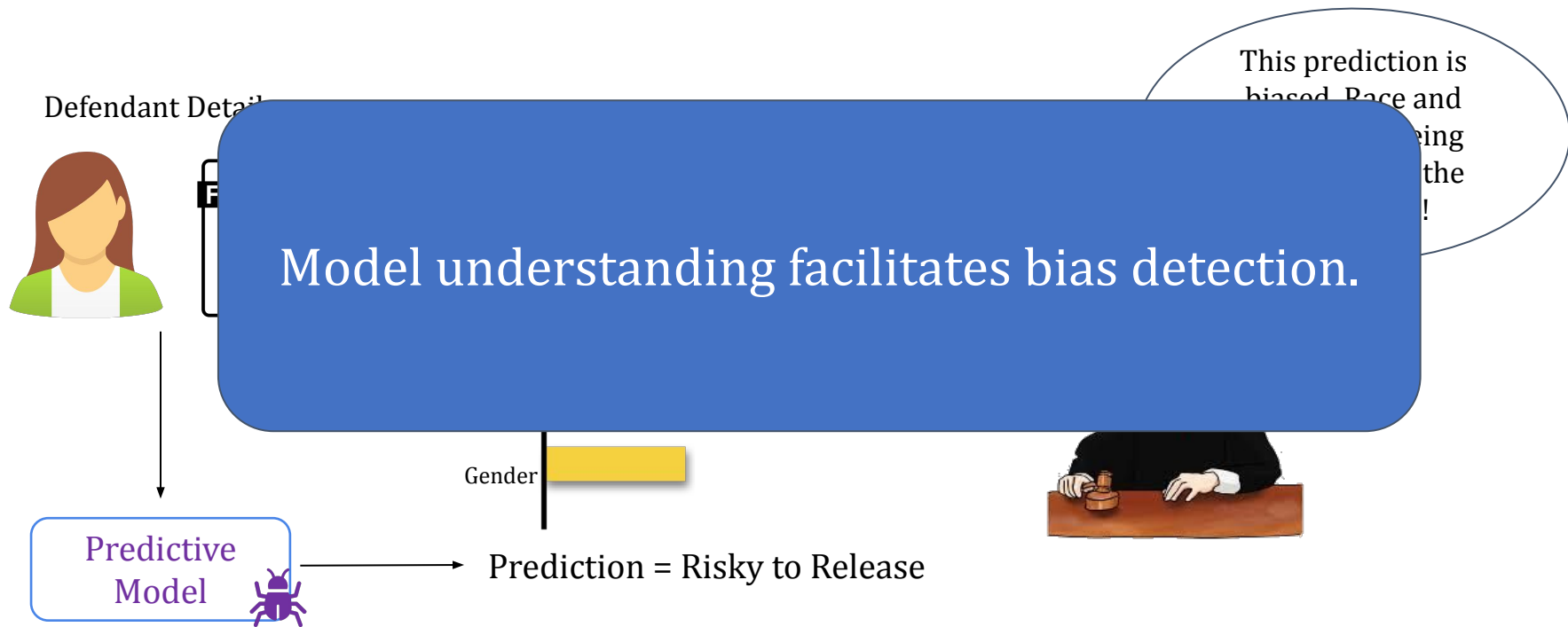


We've built a great snow detector...

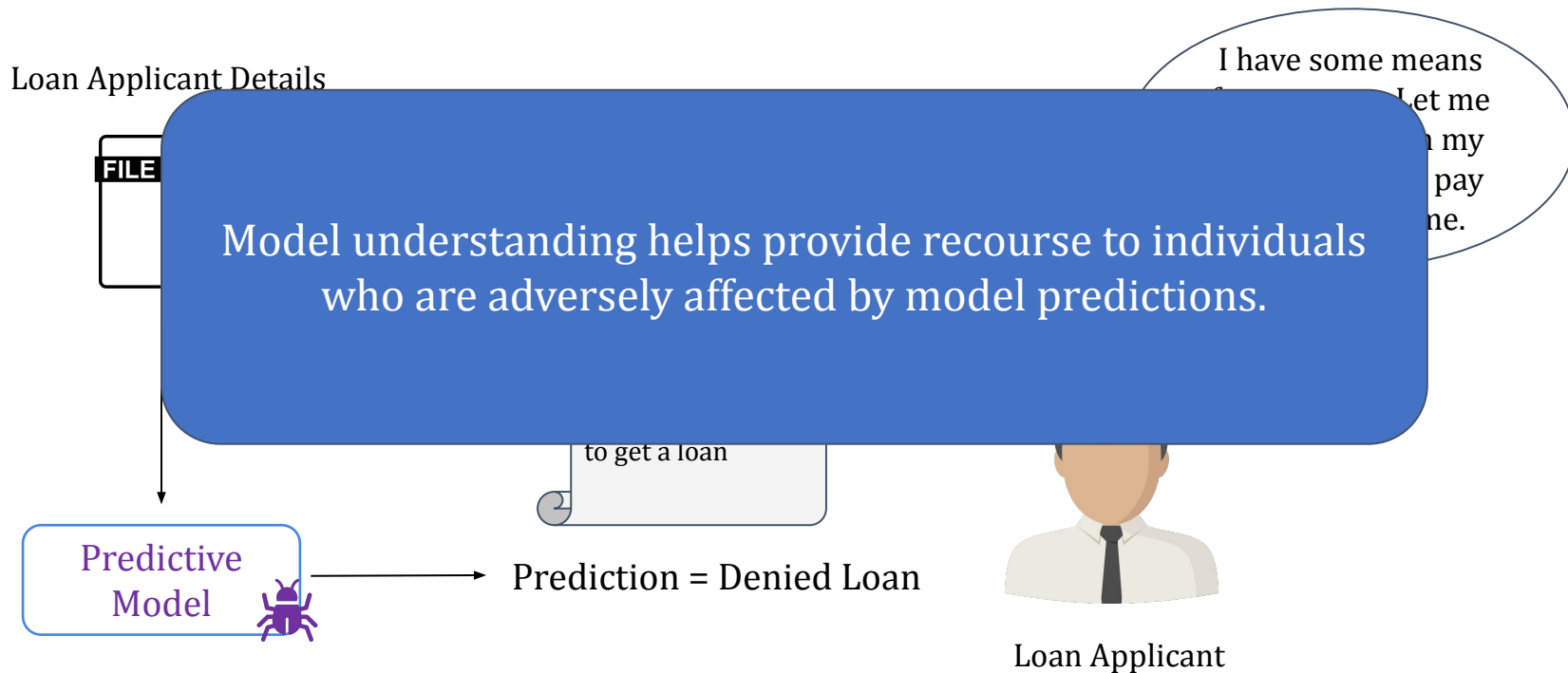
Motivation: Why Model Understanding?



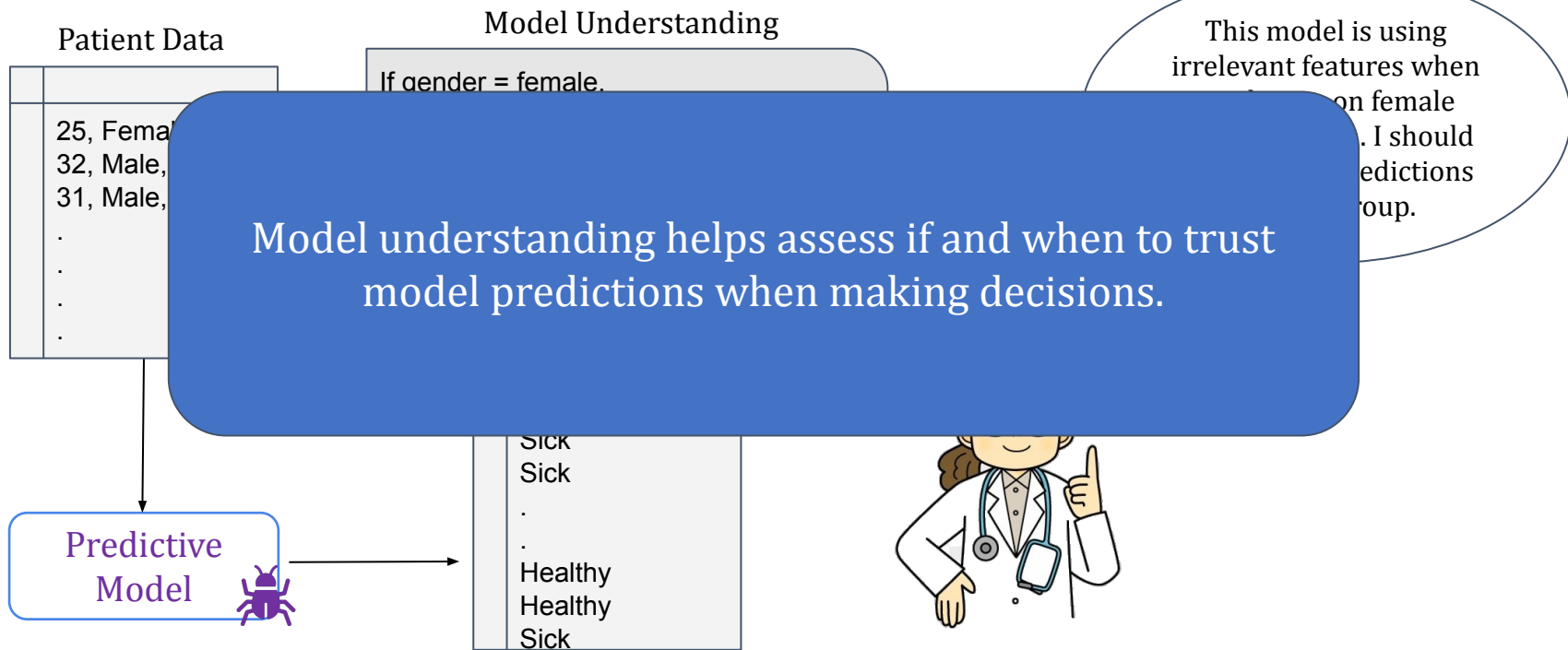
Motivation: Why Model Understanding?



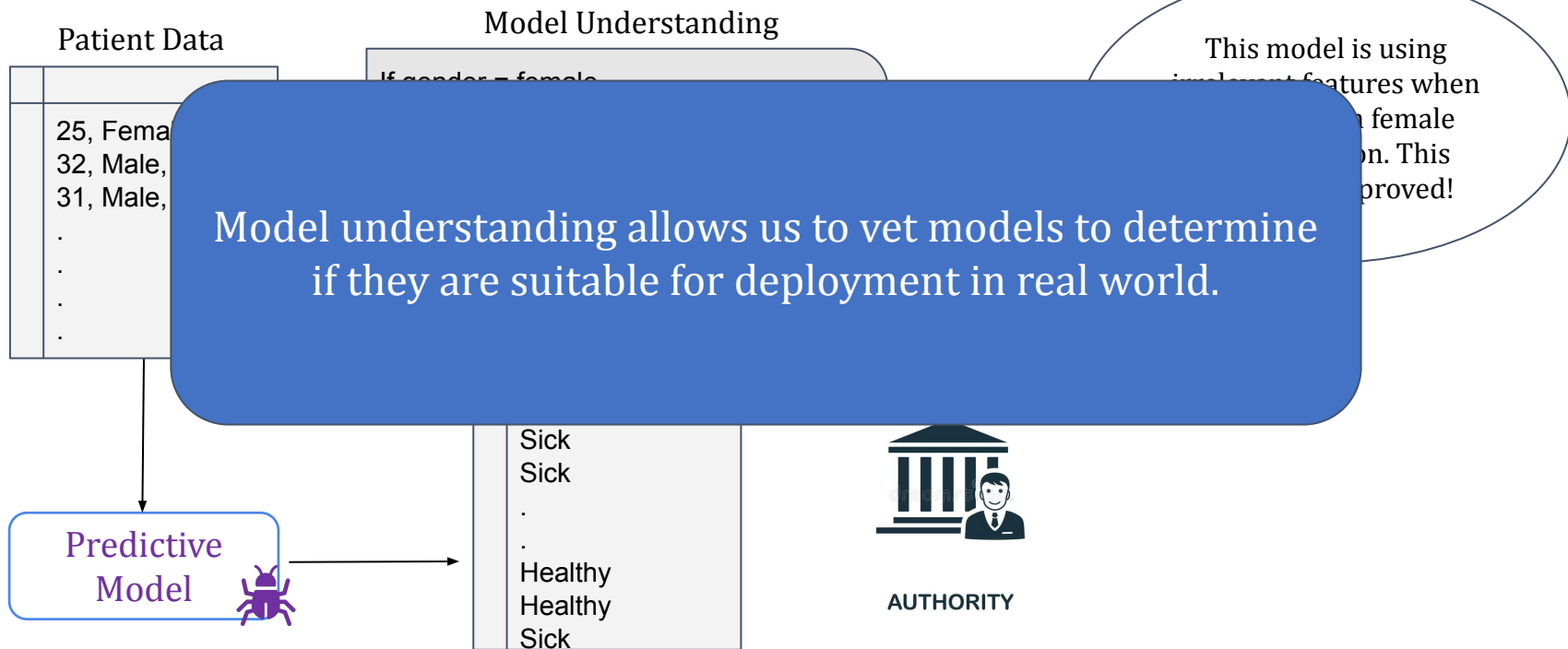
Motivation: Why Model Understanding?



Motivation: Why Model Understanding?



Motivation: Why Model Understanding?



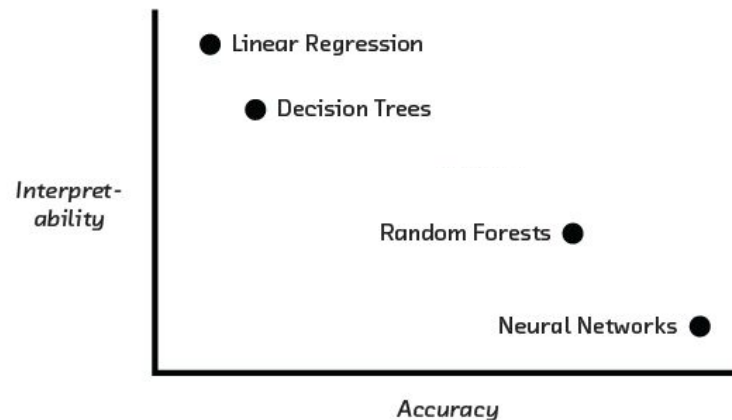
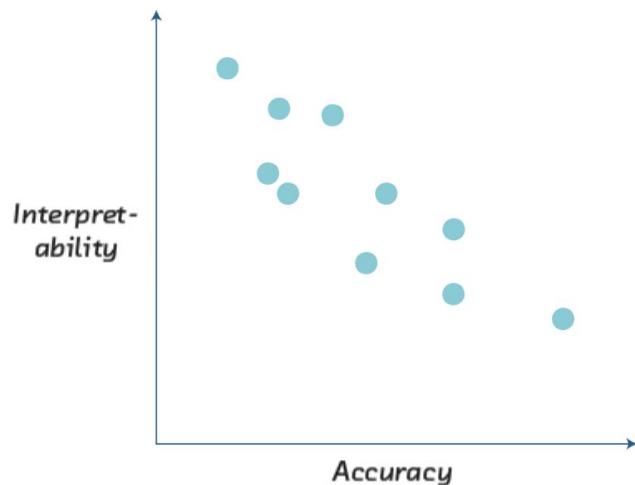
Inherently Interpretable Models

[Cireşan et. al. 2012, Caruana et. al. 2006,
Frosst et. al. 2017, Stewart 2020]

VS.

Post hoc Explanations

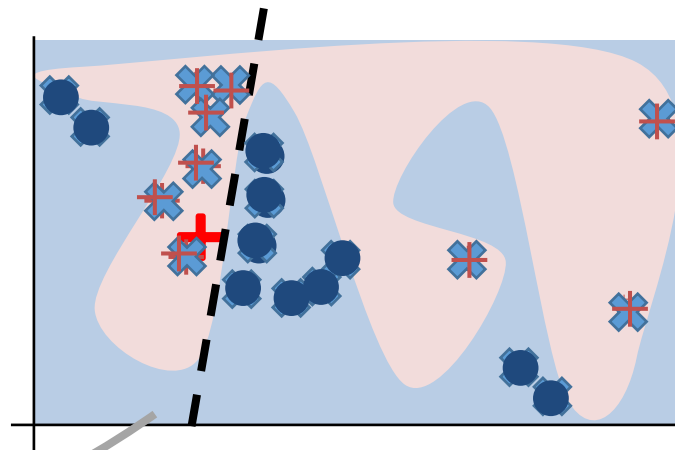
Example



In ***certain*** settings, *accuracy-interpretability trade offs* may exist.

LIME: Sparse Linear Explanations

1. Sample points around x_i
2. Use model to predict labels for each sample
3. Weigh samples according to distance to x_i
4. Learn simple model on weighted samples
5. Use simple model to explain



Integrated Gradients

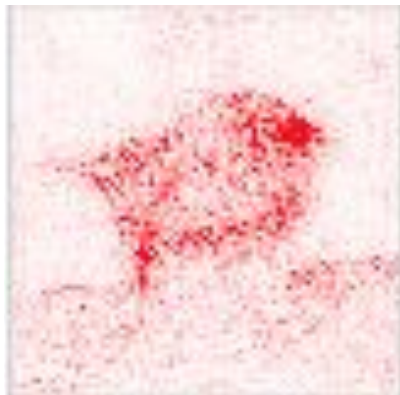
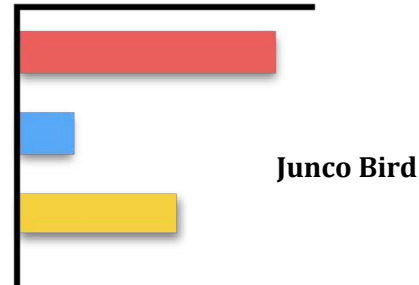
Input




Model



Predictions



Path integral: 'sum' of
interpolated gradients

$$(x - \tilde{x}) \times \int_{\alpha=0}^1 \frac{\partial F(\tilde{x} + \alpha \times (x - \tilde{x}))}{\partial x}$$


Baseline input

Comparison

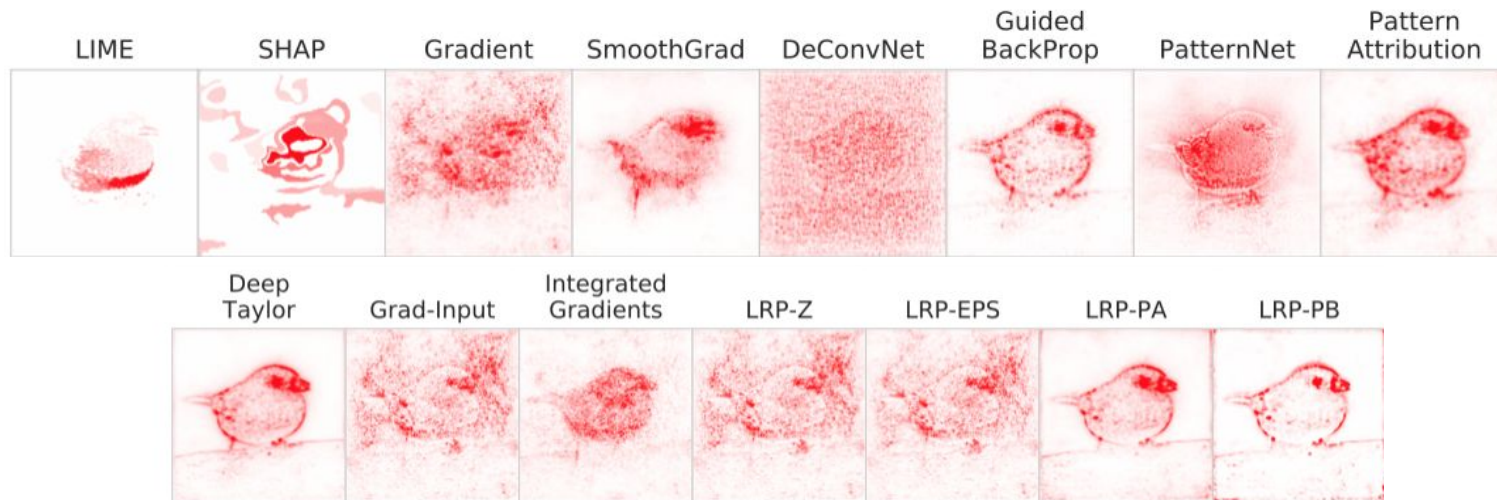
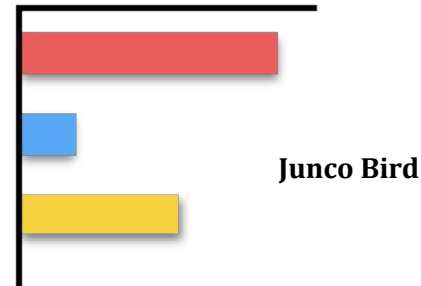
Input



Model



Predictions



/ Integrated gradients

- Let us test the integrated gradients methods on an encoder classifier
- The method is very simple, but needs a “zero” (reference, neutral, empty) input
- For images, this would be an image of white noise, or an empty image of some kind
- For text, it is harder to establish an “empty sentence”
- We can use a sequence of [PAD] symbols, for example

https://github.com/TurkuNLP/textual-data-analysis-course/blob/main/model_explainability.ipynb