Minej/bert-base-personality

like 39

Text Classification | Transformers | PyTorch | Safetensors | English | bert | Inference Endpoints | arxiv:1810.04805 | License: mit

Model card | Files and versions | Community 1

Train | Deploy | Use this model

How to Get Started with the Model

To use the model through Hosted inference API, follow the code snippet provided below:

```
from transformers import BertTokenizer, BertForSequenceClassification

def personality_detection(text):
    tokenizer = BertTokenizer.from_pretrained("Minej/bert-base-personality")
    model = BertForSequenceClassification.from_pretrained("Minej/bert-base-person

    inputs = tokenizer(text, truncation=True, padding=True, return_tensors="pt")
    outputs = model(**inputs)
    predictions = outputs.logits.squeeze().detach().numpy()

    label_names = ['Extroversion', 'Neuroticism', 'Agreeableness', 'Conscientious
    result = {label_names[i]: predictions[i] for i in range(len(label_names))}

    return result
```

Result Format

The personality_detection function returns a dictionary containing the predicted personality traits based on the given input text.

The dictionary contains the following personality traits with their corresponding predicted values:

Extroversion: A value between 0 and 1 representing the predicted extroversion trait. Neuroticism: A value between 0 and 1 representing the predicted neuroticism trait. Agreeableness: A value between 0 and 1 representing the predicted agreeableness trait. Conscientiousness: A value between 0 and 1 representing the predicted conscientiousness trait. Openness: A value between 0 and 1 representing the predicted openness trait.

```
text_input = "I am feeling excited about the upcoming event."
personality_prediction = personality_detection(text_input)

print(personality_prediction)
```

Output:

```
{
    "Extroversion": 0.535,
    "Neuroticism": 0.576,
    "Agreeableness": 0.399,
    "Conscientiousness": 0.253,
    "Openness": 0.563
}
```

Note: The values in the example output are just placeholders and may not reflect the actual predictions.

You can modify the example code and the result format to match your specific use case and desired output format.

Model Description

Transfer Learning for Big Five Personality Prediction

In machine learning, training accurate models can be challenging when labeled data is limited. Transfer learning offers a solution by leveraging pre-existing labeled data from a similar task or domain. By transferring knowledge learned from one task to another, we can overcome data scarcity and train more effective models.

In this project, we used transfer learning with the BERT BASE UNCASED model to predict Big Five personality traits. The model was fine-tuned on a curated dataset for personality traits, learning patterns between input text and personality characteristics. By applying transfer learning, we improved the accuracy of personality trait predictions.

By leveraging transfer learning and fine-tuning BERT BASE UNCASED, we accurately predict an individual's Big Five personality traits based on their input text. This approach addresses the challenges of limited labeled data in personality prediction, providing insights into individuals' personalities.

This project showcases the power of transfer learning in machine learning and highlights the effectiveness of BERT BASE UNCASED for predicting Big Five personality traits.

- **Model type:** BERT BASE UNCASED
- **Language(s) (NLP):** English
- **License:** MIT
- **Finetuned from model [optional]:** https://huggingface.co/bert-base-uncased

Uses

Direct Use

The personality prediction model can be used directly by individuals who are interested in gaining insights into their own personality traits based on their input text. Users can input text and receive predictions for the Big Five personality traits.

Downstream Use

This model is not intended for downstream use or fine-tuning for specific tasks. It is designed as a standalone personality prediction model.

Out-of-Scope Use

This model is not suitable for uses beyond personality prediction. It should not be used for making critical decisions or judgments about individuals in areas such as employment, education, or legal matters.

Bias, Risks, and Limitations

The personality prediction model, like any machine learning model, has certain limitations and potential biases that should be taken into account:

```
Limited Context:
The model makes predictions based on input text alone and may not capture the ful

Generalization:
The model predicts personality traits based on patterns learned from a specific d

Ethical Considerations:
Personality prediction models should be used responsibly, with an understanding t

Privacy Concerns:
The model relies on user-provided input text, which may contain sensitive or pers

False Positives/Negatives:
The model's predictions may not always align perfectly with an individual's actua
```

Recommendations

To mitigate risks and limitations associated with personality prediction models, the following recommendations are suggested:

```
Awareness and Education:
Users should be informed about the limitations and potential biases of the model.

Avoid Stereotyping and Discrimination:
Users should be cautious about making judgments or decisions solely based on pred

Interpret with Context:
Interpret the model's predictions in the appropriate context and consider additio

Data Privacy and Security:
Ensure that user data is handled securely and with respect to privacy regulations

Promote Ethical Use:
Encourage responsible use of personality prediction models and discourage misuse
```

It is important to note that the above recommendations are general guidelines, and further context-specific recommendations should be developed based on the particular use case and ethical considerations.

How to Download the Model

If you would like to download the model files and use them instead of the Hosted inference API, then you can follow the code snippet provided below:

```
from transformers import BertForSequenceClassification, BertTokenizer
import torch

# Initialization of the model values
model = BertForSequenceClassification.from_pretrained(".", num_labels=5)
tokenizer = BertTokenizer.from_pretrained('.', do_lower_case=True)
model.config.label2id = {
    "Extroversion": 0,
    "Neuroticism": 1,
    "Agreeableness": 2,
    "Conscientiousness": 3,
    "Openness": 4,
}
model.config.id2label = {
    "0": "Extroversion",
    "1": "Neuroticism",
    "2": "Agreeableness",
    "3": "Conscientiousness",
    "4": "Openness",
}

def personality_detection(model_input: str) -> dict:
    '''
    Performs personality prediction on the given input text

    Args:
        model_input (str): The text conversation

    Returns:
        dict: A dictionary where keys are speaker labels and values are their per
    '''

    if len(model_input) == 0:
        ret = {
            "Extroversion": float(0),
            "Neuroticism": float(0),
            "Agreeableness": float(0),
            "Conscientiousness": float(0),
            "Openness": float(0),
        }
        return ret
    else:
        dict_custom = []
        preprocess_part1 = model_input[:len(model_input)]
        dict1 = tokenizer.encode_plus(preprocess_part1, max_length=1024, padding=
        dict_custom['input_ids'] = [dict1['input_ids'], dict1['input_ids']]
        dict_custom['token_type_ids'] = [dict1['token_type_ids'], dict1['token_ty
        dict_custom['attention_mask'] = [dict1['attention_mask'], dict1['attentio
        outs = model(torch.tensor(dict_custom['input_ids']), token_type_ids=None,
        b_logit_pred = outs[0]
        pred_label = torch.sigmoid(b_logit_pred)
        ret = {
            "Extroversion": float(pred_label[0][0]),
            "Neuroticism": float(pred_label[0][1]),
            "Agreeableness": float(pred_label[0][2]),
            "Conscientiousness": float(pred_label[0][3]),
            "Openness": float(pred_label[0][4]),
        }
        return ret

personality_prediction = personality_detection(text_input)
```

Make sure you have the required dependencies installed (transformers and torch). This code snippet initializes the model, tokenizer, and configuration. It then defines the personality_detection function, which takes a text conversation as input and returns a dictionary with personality predictions for each speaker.

You can call the personality_detection function with your input text to obtain the personality predictions. The personality_prediction variable will hold the resulting dictionary.

Please note that this code assumes you have already downloaded the necessary model files (config.json, pytorch_model.bin, special_tokens_map.json, tokenizer_config.json, vocab.txt) and placed them in the current directory (indicated by "."). Adjust the paths and filenames accordingly if needed.

Citation

```
@article{DBLP:journals/corr/abs-1810-04805,
  author    = {Jacob Devlin and
               Ming{-}Wei Chang and
               Kenton Lee and
               Kristina Toutanova},
  title     = {{BERT:} Pre-training of Deep Bidirectional Transformers for Langua
               Understanding},
  journal   = {CoRR},
  volume    = {abs/1810.04805},
  year      = {2018},
  url       = {http://arxiv.org/abs/1810.04805},
  archivePrefix = {arXiv},
  eprint    = {1810.04805},
  timestamp = {Tue, 30 Oct 2018 20:39:56 +0100},
  biburl    = {https://dblp.org/rec/journals/corr/abs-1810-04805.bib},
  bibsource = {dblp computer science bibliography, https://dblp.org}
}
```

More Information

TBA

Downloads last month
4,041,901

Safetensors 1B
Model size: 109M params
Tensor type: I64 · F32

Inference API
Cold
Text Classification
Examples
I like you. I love you
Compute
View Code 0.3s
Maximize

Spaces using Minej/bert-base-personality 6