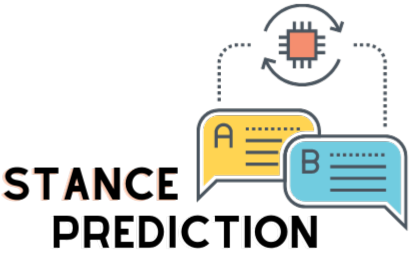
NATURAL LANGUAGE PROCESSING - PROJECT WORK REPORT

Lorenzo Pratesi, Martina Rossini, Vairo Di Pasquale
{lorenzo.pratesi2, martina.rossini3, vairo.dipasquale}@studio.unibo.it
`https://github.com/NLP-Team-Unibo/stance-prediction`

IMPLICIT STANCE PREDICTION ON POLITICAL DEBATES USING SPEECH FEATURES



Artificial Intelligence Master's Degree

Alma Mater Studiorum - University of Bologna

July 24 2022

Contents

1 Summary

The task of stance prediction involves classifying the stance of a speaker's argument towards a certain target (motion). We have already addressed the problem in which only two types of stances are available, (*supporting* or *opposing*), without explicitly considering the motion, but inferring it from the speech.

With the availability of information today, large corporations increasingly need to predict users' positions on specific topics. Although humans can assess correct stances, Machine Learning models are often not up to the task; therefore, in this project, we focused on improving our models that addressed the Stance Prediction task by trying different and new techniques.

This study still relies on IBMDebater's "Debate Speech Analysis" dataset (6), which provides both speeches and their transcriptions labeled with motions and stances.

In this work, we investigate if audio features can be useful to predict the stance of political speeches without knowing the motion in advance.

To investigate this new task we developed three different architectures:

- **MulT-based model**: combines DistilBERT (8) and wav2Vec2.0 (1) outputs with a series of MulT-based models (9).
- **BART for motion generation and stance classification**: predict the motion together with the stance of the speech by encoding the text using a BART (4) encoder and two different BART decoders to extract features for the generative and the sequence classification task respectively. Using a series of crossmodal attentions, the extracted audio features are then combined with both generative and classification decoders.
- **BART for stance classification**: uses BART (4) on textual signal and combines the outputs of its decoder with those of wav2Vec2.0 (1) with a series of crossmodal attentions.

Our best MulT-based model achieved a slightly better accuracy score on the test set (%) with respect to our previous best Multimodal model. While BART-based models produced better accuracy scores without using audio features, we obtained good results on the motion generation task and better generalization on unseen samples.

This paper summarizes all the steps that we have followed, in particular, describes the used dataset, the models created, and finally, an analysis of the obtained results.

2 Background

Before going into the explanation of our additions, we consider it important to make a quick description of the previous models from which we started and briefly give a background on the new models and technologies that we used.

2.1 Data Preprocessing

The dataset for this work is still the IBM Debater 'Debate Speech Analysis' dataset (6): it contains a set of speeches discussing a single motion and they can be either *supporting* – meaning that the speaker is arguing in favor of the topic of interest – or *opposing*, meaning that the speaker is arguing against the motion. Each speech is also accompanied by its textual transcription, which has on average a length of 738 tokens.

While we previously used the DistilBERT tokenizer from HuggingFace `transformers` library (10) to extract a word sequence from a string, we now switched to the BART tokenizer, which was developed specifically to work with BART-inspired models. It's important to note that, differently from BERT and DistilBERT, BART doesn't rely on a specific `[CLS]` token to perform sentence classification, but usually employs the last element of its decoder's terminal hidden state.

Similarly to what was done for our previous project, believing that the first (or last) seconds of the wavelength are enough for the models to gather some information regarding the intonation, rhythm, and stress of the speech and thus we chose to truncate each of them to just 15 or 10 seconds of the wavelength..

2.2 Recap: Previous System Description and Results

As a result of our previous work, we created three models based on text and audio data, with a third model combining the two. We were primarily interested in finding out whether multimodal information could improve the performance of our models.

- **Text model**: tries to predict the stances of the speech transcriptions. Its core is DistilBERT (8).
- **Audio model**: tries to predict the stances of the speeches. Its core is wav2vec 2.0 (1).
- **Multimodal model**: tries to predict the stances using both speeches and their transcriptions. It combines both the aforementioned models.

We evaluated our models using accuracy as a metric. The evaluations on the test set showed that the best Text model achieved 93.82% accuracy, the best Audio model achieved 92.04% accuracy and the best Multimodal model achieved a 94.65% accuracy, showing a little improvement in using both audio and text signals.

2.3 BART

BART (4) is a state-of-the-art denoising autoencoder for pretraining sequence-to-sequence models. BART is trained by corrupting text with an arbitrary noising function and learning a model to reconstruct the original text. It uses a standard Transformer-based neural machine translation architecture which, despite its simplicity, can be seen as generalizing BERT (3) (due to the bidirectional encoder), GPT (2) (with the left-to-right decoder), and other recent pretraining schemes. BART is particularly effective when fine-tuned for text generation but also works well for comprehension tasks.

2.4 MulT - Multimodal Transformer

Combining time-based signals having different natures, like text and speech signals is not straightforward. Generally, they come unaligned, so it is hard to capture their relationship. Signals can also have long-term dependencies and extracting them could be challenging. To address those issues, Yao-Hung et al. presented the Multimodal Transformer (MulT) (9), a Transformed-based architecture with crossmodal attention mechanisms. Like a Transformer block, MulT can capture long-term dependencies of signals, while also trying to align different signals with its crossmodal attention. A single stack of MulT allows capturing the oriented relationship between signals, i.e. from speech to text but not vice versa.

3 System Description

We developed three different architectures to investigate the audio contribution to the implicit stance prediction task. The first tries to combine the outputs of the DistilBERT and wav2vec2.0-based models (described in our previous project) with a series of MulT. The second tries to predict the motion together with the stance of the speech; it encodes the text using a BART encoder and uses two different BART decoders to extract features for the generative and the sequence classification task respectively. The extracted audio features are then joined together with both the generative and classification decoders using again a series of crossmodal attentions. The third uses BART on textual signal and combines the outputs of its decoder with those of wav2Vec2.0 with a series of crossmodal attentions.

In order to combine audio and text signals, we developed a custom BART Decoder: it simply executes the original BART decoder and then feeds its output into either a multi-head attention layer or a sequence of Multimodal Transformer layers. When using the audio signal, the multi-head/crossmodal attentions of this architecture use the decoder input ids as the query and the audio embeddings produced by wav2vec2.0 as the key and value. On the other hand, when the audio signal is not provided - meaning that we are not in a multimodal setting - the decoder input ids are used for both query and key/value, thus performing a self-attention step.

3.1 MulT-based model

Following our previous work, we defined the TextModel and the AudioModel, for the extraction of textual and audio features respectively. To combine them, we experimented with some MulT settings:

- **Text2Audio**: with this setting, the MulT takes as queries the audio embeddings and textual key/values. In this way, we condition the audio features with textual contextualized embeddings.
- **Audio2Text**: with this setting, the MulT takes as queries the textual embeddings and audio key/values. In this way, we condition the text features with audio contextualized embeddings.
- **Signal2Signal**: in this setting, we combine the above settings.

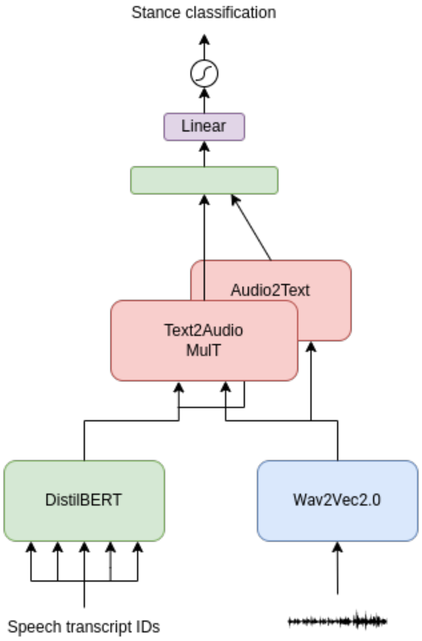


Figure 1: MulT-based model

Every MulT has 4 Transformer layers with cross-modal attention. Before classification, we experimented with different pooling strategies on the output embeddings of MulT, i.e. taking only the first/last embedding of the sequence or its average. Then a classification head is attached to this reduction. With the Signal2Signal setting, before classification, we concatenate together the reductions of each MulT and then we classify..

3.2 BART for motion generation and stance classification

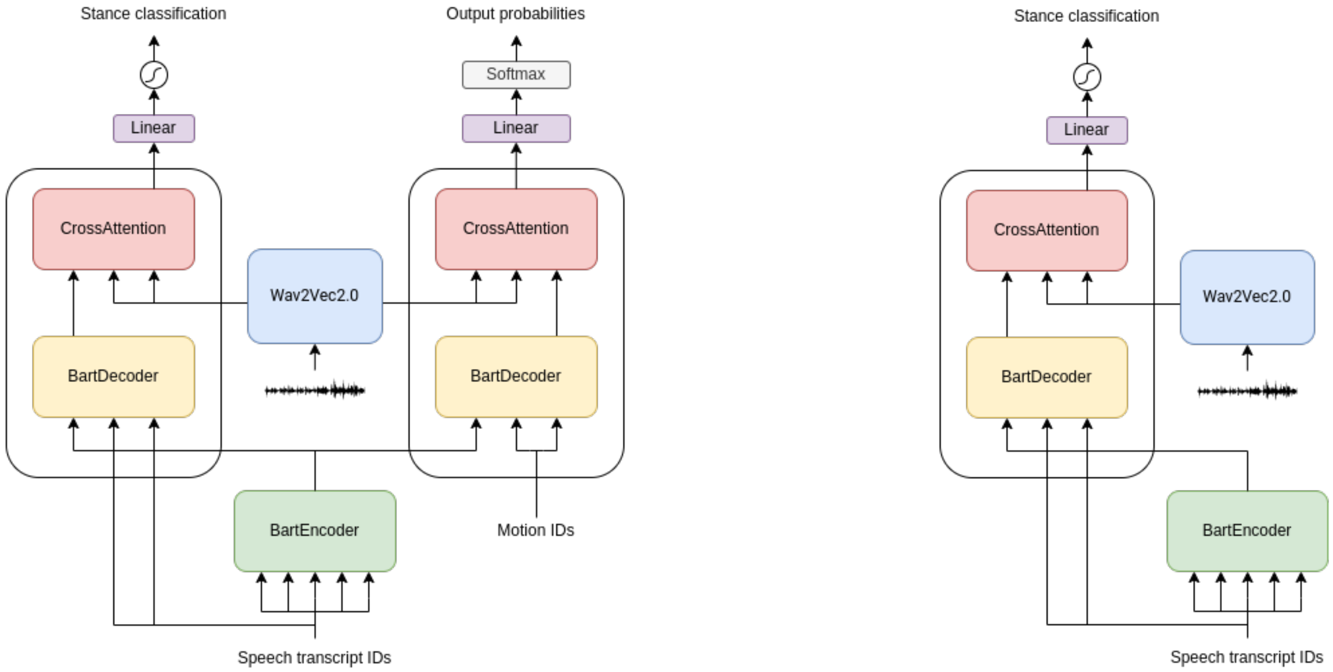
This architecture consists of one BART original encoder and two custom decoders, for the sequence classification and motion generation respectively. The encoder is defined exactly as in the original paper. The encoder takes the tokenized transcriptions as input and it is shared between the two decoders. Its outputs are fed into both decoders as queries and also as keys and values only for the classification decoder. Instead, the generation decoder takes as keys and values the tokenized motion. A classification head is attached to the last embedding of the outputs of the classification decoder, with a sigmoid activation at the end of it. Instead, a language model head, consisting of a linear layer with a softmax activation, is used for the prediction of the next token for the generation decoder. If the audio is used, its embeddings, computed by wav2vec2.0, are fed into each custom decoder as stated before.

3.3 BART for stance classification

This architecture was developed with the intent of creating a fair baseline against which we could compare our multi-task BART model: it comprises a BART encoder - defined exactly like in the original paper - followed by our custom BART decoder and by a classification head. The decoder takes as input the encoder's output and the tokenized text is shifted to the right by one token: the final hidden state of the last decoder token contains then a representation for the complete input and can be used for sentence classification.

The classification head constitutes dropout and ReLU activation applied to the last element of the decoder's last hidden state, plus a linear layer that produces the score itself.

Figure 2: BART-based models



(a) BART for motion generation and stance classification

(b) BART for stance classification

4 Experimental Setup and Results

We are now going to describe the experimental setup used to train each architecture described before. In all the experiments, we used the provided validation set to benchmark the training procedure. The loss function used to train the classification architectures is the binary cross-entropy, while for the generation architectures we used the categorical cross-entropy loss. In our multi-task architectures, the loss is just the sum of the classification and generation losses stated before. We optimize using Adam with starting learning rate $\alpha = 2.0 \exp \text{-}5$ and with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ used to compute the running averages of the gradient and its square respectively. For all the architectures, we employed some dropout layers with variable values and an early stopping criterion to add regularization.

In all the experiments, we selected different cut strategies of the text and, when used, for the audio:

- **Cut first**: take only the first 512 tokens of the transcription and only the first 10 seconds of the recorded speech.
- **Cut last**: take only the last 512 tokens of the transcription and only the last 10 seconds of the recorded speech.
- **Cut both**: take only the first 256 and last 256 tokens of the transcription and concatenate them, following the temporal order. Also, only take the first 5 and the last 5 seconds from the speech recording and concatenate them in a temporal order.

4.1 MulT-based model

For all the experiments, we trained only the last 2 Transformer layers of DistilBERT and all the Multimodal Transformers. We experimented with all the sample cut strategies and all the possible MulT combination settings described in Section 3.1 with average pooling. Then on the best resulting models, we tried the remaining pooling strategies to see the best one. All the experiments are shown in Table 1

Table 1: Hyperparameters for MulT-based model

Model Name	Sample Cut	Pooling Op	Cross Type
cross_text_cut_first	first	avg	audio2text
cross_text_cut_last	last	avg	audio2text
cross_text_cut_both	both	avg	audio2text
cross_audio_cut_first	first	avg	text2audio
cross_audio_cut_last	both	avg	text2audio
cross_audio_cut_both	both	avg	text2audio
cross_both_cut_first	first	avg	both
cross_both_cut_last	last	avg	both
cross_both_cut_both	both	avg	both
cross_both_cut_both_pool_first	both	first	both
cross_both_cut_both_pool_last	both	last	both

4.2 BART for motion generation and stance classification

For all our experiments we trained the whole BART architecture, only keeping fixed the Embedding layer. We tried both a multimodal version (i.e.: passing the audio chunks to wav2vec2.0 and using the produced embeddings as inputs for our custom BART decoders) and an only-text version. Additional parameters, like the number of Multimodal Transformer layers in the custom BART decoders, are shown in Table 2. Note that if the number of MulT Transformer layers is zero, then it means that the custom decoder is using just the mutlihead attention. In the multimodal version, we always used a version of wav2vec2.0 with 12 Transformer layers, all of which were fine-tuned. We experimented with two variants of cross-modal attention, one using MulT with two layers and the other using multi-head cross attention with 8 heads. All the obtained results are shown in Table 5.

Table 2: Hyperparameters for multi-task BART

Model Name	Multimodal	Dropout	# Mult Layers
default	✓	0.3	0
default_noaudio		0.3	0
cross_2	✓	0.3	2
cross_2_high_drop	✓	0.4	2
cross_2_noaudio		0.3	2

4.3 BART for stance classification

For all our experiments we trained the whole BART architecture, only keeping fixed the Embedding layer. Similar to what we did for the multi-task model above, we tried both a multimodal and a text-only version. Additional parameters, like the number of Multimodal Transformer layers in the custom BART decoder, are shown in Table 3. Note that if the number of MulT Transformer layers is zero, then it means that the custom decoder is using just the multi-head attention. In the multimodal version, we always used a version of wav2vec2.0 with 12 Transformer layers, all of which were fine-tuned. The results we obtained on the test and validation splits are shown in Tbale 6.

4.4 Results

Since our dataset doesn't suffer from class imbalance, we evaluated all our stance prediction models using accuracy. We also analyzed the confusion matrix as well as some other metrics like precision, recall, and F1-score, but we did not monitor them during training, nor did we use them to take decisions.

Table 3: Hyperparameters for BART for stance classification

Model Name	Multimodal	Dropout	# Mult Layers
default	✓	0.3	0
default_noaudio		0.3	0
default_noaudio_high_drop		0.4	0
cross_2	✓	0.3	2
cross_2_noaudio		0.3	2

Regarding those models that also perform text generation, we choose to evaluate their performances using two widely common metrics, BLEU and ROUGE. BLEU stands for Bilingual Evaluation Understudy score and it was originally developed as a metric for machine translation (7) but it has since become widely used for the evaluation of text-generation models in general; it's interesting to note that this score also applies a brevity penalty, meaning that there is a penalty when the generated text is smaller compared to the target text. ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation (5) and is a set of metrics that were originally introduced as for automatic evaluation of text summarization tasks. In particular, we used:

- ROUGE-1: unigram-based score between generated text and label
- ROUGE-2: bigram-based score between generated text and label
- ROUGE-L: score based on the Longest Common Subsequence (LCS); the intuition is that the longer the LCS of two generated sentences is, the more similar they are.

Table 4: MulT-based models: validation and test results

Model Name	Validation Accuracy	Test Accuracy
cross_text_cut_first	97.71%	94.65%
cross_text_cut_last	96.42%	92.18%
cross_text_cut_both	97.85%	94.51%
cross_audio_cut_first	97.85%	93.00%
cross_audio_cut_last	97.28%	92.87%
cross_audio_cut_both	98.13%	92.59%
cross_both_cut_first	98.13%	93.82%
cross_both_cut_last	96.56%	93.14%
cross_both_cut_both	**98.28**%	94.59%
cross_both_cut_both_pool_first	**98.28**%	94.23%
cross_both_cut_both_pool_last	**98.28**%	**94.78**%

Table 5: BART for motion generation and stance classification: validation and test results

Model Name	Validation			Test		
	Accuracy	BLEU	ROUGE-L	Accuracy	BLEU	ROUGE-L
default	98.57%	0.0	1.83	94.65%	0.0	1.94
default_noaudio	98.85%	0.0	0.009	94.79%	0.0	0.003
cross_2	98.85%	62.01	82.87	**95.34**%	56.16	79.67
cross_2_high_drop	**99.28**%	70.14	87.67	94.92%	67.24	85.83
cross_2_noaudio	98.57%	69.78	87.73	95.06%	66.18	84.83

Table 6: BART for stance classification: validation and test results

Model Name	Validation Accuracy	Test Accuracy
default	97.56%	93.82%
default_noaudio	98.71%	**95.88**%
default_noaudio_high_drop	98.57%	93.83%
cross_2	98.28%	94.92%
cross_2_noaudio	**98.99**%	95.06%

5 Discussion

Looking at the results in Table 4, we can immediately see that the best results, on both test and validation data, are obtained by models which use the `both` sample cut strategy; this makes sense, as we noticed in our previous experiments that the last part of an argumentative speech is quite important when performing stance prediction as that's where the speaker draws his conclusions. However, analyzing our dataset, we also noticed that a lot of the debaters started their speech by repeating the motion they were given: this means that the first sentence of our examples can be extremely useful for our task, since we are not explicitly passing the motion to our models. Notice that Table 4 also shows that the pooling strategy does not have a huge impact on performance, even though the `avg` strategy seems to help the model generalize better on unseen examples, as it provides a better accuracy on the test set.

Notice that all the experiments done for BART (both for stance classification only and for multi-task) were done using the `both` sample cut strategy because of what we pointed out above regarding the MulT-inspired model's results. From Table 5 we can see how the models that do not use the Multimodal Transformer to incorporate textual and audio features perform poorly at the text generation task, while retaining good classification performances. On the other hand, adding just two Multimodal Transformer layers to the end of the BART decoder results in impressive generative performances, where the models are able to re-create the motion almost exactly. Moreover, we can see that using the audio information seems to improve the generation capabilities of our model (i.e.: the results of `cross_2_high_drop` are slightly better than those of `cross_2_noaudio`); the classification performance remains quite similar instead.

Finally, comparing the results in Tables 5 and 6 we can see that the generative task doesn't always seems to improve performance and that using acoustic information sometimes seems to hurt performance: we should however keep in mind that adding the generative task implies adding approximately 120 million parameters (those of our custom BART decoder, plus those of the classification head), while adding audio embeddings implies adding wav2vec2.0 to our overall architecture. Indeed, while the IBM Debater 'Debate Speech Analysis' dataset is quite big, we still run the risk of overfitting, which could explain some of our inconclusive results.

Regarding the two common error types our previous models made, we can say that using the `both` sample cut strategy seems to drive down the number of errors due to limiting the input tokens to just 512. Similarly, introducing the motion generation task seems to improve the classifier's capability to correctly predict stance for all those motions that were formulated in a negative way. However, in the multi-task model we can identify at least one more category of common errors: indeed, we often have a wrongly predicted stance is correlated with a wrong motion generation. This happens in particular with motions containing phrases like "more good than harm"/"more harm than good", which our systematically cause confusion for our generative model. A couple of examples of this kind of error are shown in Table 7. Errors on motions containing these two phrases amount to about 45% of our errors, while the others are mostly due to the generative model not catching the correct polarity of the motion (i.e.: inserting a "not" in an originally positive motion or vice versa), which consequently leads the classification head astray.

Table 7: Generative errors: examples

Target Motion	Predicted Motion	Target Class	Predicted Class
Private universities bring more good than harm	Private universities bring more harm than good	pro	con
Abstinence-only sex education brings more good than harm	We should ban abstinence	con	pro
International adoption brings more good than harm	International adoption brings more harm than good	con	pro

References

[1] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *CoRR*, abs/2006.11477, 2020.

[2] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[4] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019.

[5] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[6] Matan Orbach, Yonatan Bilu, Assaf Toledo, Dan Lahav, Michal Jacovi, Ranit Aharonov, and Noam Slonim. Out of the echo chamber: Detecting countering debate speeches. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7073–7086, Online, July 2020. Association for Computational Linguistics.

[7] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[8] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019.

[9] Yao-Hung Hubert Tsai, Shaojie Bai, Paul Pu Liang, J. Zico Kolter, Louis-Philippe Morency, and Ruslan Salakhutdinov. Multimodal transformer for unaligned multimodal language sequences. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Florence, Italy, 7 2019. Association for Computational Linguistics.

[10] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771, 2019.