Join our book community on Discord

https://packt.link/EarlyAccessCommunity

Transformers reveal their full potential when we unleash pretrained models and watch them perform downstream **Natural Language Understanding (NLU)** tasks. It takes a lot of time and effort to pretrain and fine-tune a transformer model, but the effort is worthwhile when we see a multi-billion parameter transformer model in action on a range of NLU tasks.Advanced NLP models have achieved the quest of outperforming the human baseline. The human baseline represents the performance of humans on NLU tasks. Humans learn transduction at an early age and quickly develop inductive thinking. We humans perceive the world directly with our senses. Machine intelligence relies entirely on our perceptions transcribed into words to make sense of our language. Yet, machine intelligence has now surpassed basic human baselines.We will first bridge the gap between the functional and mathematical architecture of transformers we went through in *Chapter 2*, *Getting Started with the Architecture of the Transformer Model*, by introducing *emergence*. All machine models possess emergence abilities and the capacity to learn new things independently, but a transformer's unique architecture has taken language understanding to another level.We will then see how to measure the performance of transformers. Measuring **Natural Language Processing (NLP)** task performance remains a straightforward approach involving accuracy scores in various forms based on true and false results. These results are obtained through benchmark tasks and datasets. SuperGLUE, for example, is a wonderful example of how Google DeepMind, Facebook AI, the University of New York, the University of Washington, and others worked together to set high standards to measure NLP performances.Finally, we will explore several downstream tasks, such as the **Standard Sentiment TreeBank (SST-2)**, linguistic acceptability, and Winograd schemas.Transformers are rapidly taking NLP to the next level by outperforming other models on well-designed benchmark tasks. Alternative transformer architectures will continue to emerge and evolve.This chapter covers the following topics:

- How transformer attention heads produce outputs
- Measuring transformer performance versus Human Baselines
- Measurement methods (Accuracy, F1-score, and MCC)
- Benchmark tasks and datasets
- SuperGLUE downstream tasks
- Linguistic acceptability with CoLA
- Sentiment analysis with SST-2
- Named Entity Recognition
- Winograd schemas

Let's start by understanding how machines represent language.

The Paradigm shift: What is an NLP Task?

ChatGPT stunned the world when it suddenly became mainstream in late 2022 and early 2023. An AI could generate human-like text on practically any topic. Thousands of tasks were submitted to this incredible generative AI transformer. ChatGPT Plus, GPT-4, seemed to be able to perform any task an end-user came up with.However, OpenAI couldn't have possibly pretrained ChatGPT on thousands of tasks that could not be guessed beforehand. Nor could OpenAI have possibly fine-tuned its GPT models for everything the end-user was coming up with.Of course, a transformer model can be trained for specific tasks and determined downstream tasks such as summarizing. However, models such as ChatGPT can perform downstream tasks for which they were not trained for. This section takes us inside the head of a transformer model to see how the architecture described in *Chapter 2*, *Getting Started with the Architecture of the Transformer Model,* applies to downstream tasks. In Chapter 2, we went through the architecture of the Original Transformer. We specifically went through the attention heads starting section *Sublayer 1: Multi-head attention:*

1. The scores displayed are the result of the Q * K matrix multiplication.
2. These raw scores are scaled by dividing them by the square root of the dimension of the key vectors.
3. Then, a softmax function is applied to the raw scores, to sum up to 1.

If necessary, take your time to go through Chapter 2 again. How can these mathematical functions become the incredible outputs we obtain when we dialog with ChatGPT, for example?To understand how this apparent miracle happened, we must dive into the unseen depths of transformers.**Inside the head of the attention sublayer of a transformer** *In this section, the scores displayed are the output of the attention heads after the softmax but before the probabilities create a weighted sum of the value vectors(V).*This section aims to bridge the gap between understanding the mathematics of the architecture and how the model produces tokens(pieces of words or words) that become sequences of words.We will go down to code level in *Chapter 9, Shattering the Black Box with Interpretable Tools*, including BertViz_Interactive.ipynb, used for this section to illustrate how numbers become words. You do not need to run any code but to grasp how we go from mathematical probabilities to words. The key focus here is understanding the quantum leap from seemingly meaningless numbers to organized language.We will see how meaning *emerges* as a transformer model learns more and more language sequences. We will now examine the activity of attention heads through the following sentence to see the inner workings of the transformer:

Copy Explain

```
"Transformers possess surprising emerging features."
```

The sequence runs through a transformer model's sub-layer layers and reaches the attention heads as described in Chapter 2, section *Sublayer 1: Multi-head attention*.We will look at the outputs of the heads obtained without going into the mathematical calculations of the attention heads. We will not look at the code either. This can wait until Chapter 9. *We will focus on how the Original Transformer was designed to produce its disruptive outputs just like the original designers did.*Once we bridge the gap between numbers and words, we will have the understanding required to run the notebooks throughout the book.Let's focus on the outputs of the attention head.First, we will look into layer 1 and attention head 1:

Copy Explain

```
selected_layer = 1
selected_head = 1
```

The transformer analysis tool displays what the head is "thinking":

	transformers	possess	surprising	emerging	features
transformers	0.793087	0.037714	0.027054	0.020088	0.016091
possess	0.383139	0.012022	0.530375	0.007978	0.001052
surprising	0.846634	0.000771	0.026575	0.083948	0.007271
emerging	0.677940	0.001957	0.003967	0.010203	0.155751
features	0.432853	0.000938	0.001393	0.000413	0.000555

Figure 3.1: Probability table for word pairs for attention head 1 in layer 1

Each line of the chart contains a word in the sentence. Each column also includes each word in the sentence. The intersection represents the probability that the two words are related.You can see that what we call "learning" or "training" is the probability that one word is related to another in a given context (sequence).For this experiment, we will focus on the relationship between the words *transformers* and *possess*:

transformers

The attention head's "mind" is running at full speed. It is looking at the relationship between every word and every other word.It finds a high score for the *transformers-transformers* pair = 0.79 This score is not very interesting because a word is obviously related to itself.

possess

This score between the words transformer and possess is 0.037. That's not very interesting yet, either.

Copy Explain

```
selected_layer = 1
selected_head = 6
```

Now, let's look at another head in layer 1.

	transformers	possess	surprising	emerging	features
transformers	0.702030	0.122793	0.018648	0.075400	0.032200
possess	0.613370	0.102932	0.043511	0.169905	0.056358
surprising	0.522166	0.312261	0.021874	0.032769	0.079888
emerging	0.553818	0.116540	0.008703	0.031475	0.014774
features	0.713933	0.111162	0.012219	0.022824	0.011651

Figure 3.2: Probability table for word pairs for attention head 6 in layer 1

Head #6 creates different relationships.

transformers

It finds a high score for the *transformers-transformers* pair but lower than head 1 = 0.70

possess

This score between the words transformer and possess is higher than in head 1= 0.12. Now, let's jump to layer 6, head 1:

Copy Explain

```
selected_layer = 6
selected_head = 1
```

	transformers	possess	surprising	emerging	features
transformers	0.031646	0.188067	0.538230	0.009978	0.026219
possess	0.036634	0.034815	0.069137	0.043544	0.027010
surprising	0.112891	0.512405	0.114949	0.027858	0.026199
emerging	0.031653	0.086987	0.033904	0.032473	0.031259
features	0.014392	0.043559	0.029788	0.020789	0.015035

Figure 3.3: Probability table for word pairs for attention head 1 in layer 6

Head #1 is beginning to see the light.

transformers

It finds a lower score for the *transformers-transformers* pair than head 1 = 0.03It is learning that it must associate with words other than itself.

possess

This score between the word transformer and possess is higher than at layer 1= 0.18. Notice the high score of the word surprising 0.53.Finally, let's look into head layer 6, head 6:

```
selected_layer = 6
selected_head = 6
```

	transformers	possess	surprising	emerging	features
transformers	0.009803	0.200566	0.004300	0.007951	0.007760
possess	0.005146	0.007327	0.122820	0.004412	0.001436
surprising	0.007802	0.016328	0.028438	0.397363	0.174339
emerging	0.001280	0.000809	0.001532	0.029902	0.693638
features	0.018621	0.001600	0.002361	0.009339	0.020869

Figure 3.4: Probability table for word pairs for attention head 6 in layer 6

Head 6 has its own views:

transformers

It finds a lower score than head 1 for the *transformers-transformers* pair = 0.009

possess

This score between the word transformer and possess is higher than at head 1= 0.2. Notice the high score of the word surprising is lower than possess and lower than for head 1= 0.004.The training is not over yet, but we draw some insights.

1. 1. In layer 6, head 6, the model progressively finds that it should choose the word *possess*. A noun + verb is a good choice. The training is not over, but it's a good beginning.
2. 2. Suppose the training was on a masked word and that the transformer model had to find the missing word in this sentence:

```
Transformers _______ surprising emerging features.
```

The model found an emerging pattern and did learn to choose the verb possess.

1. 3. The model is thus *emergent* as many machine learning algorithms.
2. 4. The extra magic that gives a whole new meaning to *emergence* of the transformer model is that it is learning an unlimited amount of additional information!

If we look at layer 6 and head 6 for the word *transformer*, we see:

	transformers	possess	surprising	emerging	features
transformers	0.009803	0.200566	0.004300	0.007951	0.007760

Figure 3.5: Probability table for word pairs for "transformers" for attention head 6 in layer 6

If we sort them, omitting transformers, since it's the first word we are examining, see that transformers will choose: "transformers possess emerging features (surprising)."This is a good progression because we are just in the middle of the training process.If the goal was to find possess, we could say that far more information is *emerging*.Imagine when the model is learning millions of relationships with billions of parameters!We can now see how a transformer goes from numbers representing tokens (pieces of words or words) to intelligible sequences.Let's see emergence in action with ChatGPT.

Exploring Emergence with ChatGPT

Programming languages are languages, also. If we repeated the process of looking into the "head" of a transformer for source code, the transformer would also be *emergent*.The model would behave exactly like any other language sequence and produce scores for each layer and head.After being trained on the source code of GitHub as OpenAI GPT models were, it could produce lines of code. Also, since it understands language very well, it could comment the code.That's precisely what happened and what OpenAI did!Open From_training_to_emergence.ipynb in `Chapter03` of the GitHub repository of this book.The program was written, commented, and explained by ChatGPT Plus, GPT-4 only on https://openai.com/.I wrote the instructions:**Denis :** `Write a k-nearest neighbor program in Python that uses 1000 random variables using a k-NN decision boundary, 5 classes and display the result with MatPlotlib:`ChatGPT Plus, GPT-4 wrote some preliminary explanations:**GPT-4 explains(excerpt):**`The k-NN neighbors boundary is not a single, well-defined boundary but rather a collection of irregular and potentially disjoint boundaries, which can be complex and highly dependent on the specific dataset and the choice of 'k'. When 'k' is small, the decision boundaries can be quite sensitive to noise, leading to overfitting. On the other hand, when 'k' is too large, the decision boundaries may become too smooth, leading to underfitting. Choosing the optimal value of 'k' is crucial for achieving good performance with the k-NN algorithm. .../...`Then ChatGPT Plus, GPT-4 wrote the code:

Copy Explain

```
!pip install scikit-learn
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_blobs
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from matplotlib.colors import ListedColormap
# Generate 1000 random data points with 5 classes
X, y = make_blobs(n_samples=1000, centers=5, random_state=42)
# Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)
# Create a k-NN classifier with k=5
knn = KNeighborsClassifier(n_neighbors=5)
# Train the classifier with the training data
knn.fit(X_train, y_train)
# Create a mesh to plot the decision boundaries
h = .02  # step size in the mesh
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
# Predict the mesh points' class labels
Z = knn.predict(np.c_[xx.ravel(), yy.ravel()])
# Put the result into a color plot
Z = Z.reshape(xx.shape)
cmap_light = ListedColormap(['#FFAAAA', '#AAFFAA', '#AAAAFF', '#AFAFAF', '#FFD700'])
cmap_bold = ListedColormap(['#FF0000', '#00FF00', '#0000FF', '#808080', '#FFA500'])
plt.figure()
plt.pcolormesh(xx, yy, Z, cmap=cmap_light)
# Plot the training points
plt.scatter(X_train[:, 0], X_train[:, 1], c=y_train, cmap=cmap_bold, edgecolor='k',
s=20)
plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())
plt.title("5-Class classification (k = 5, n_samples = 1000)")
plt.show()
```

I copied the program into Google Colab and obtained an excellent plot:

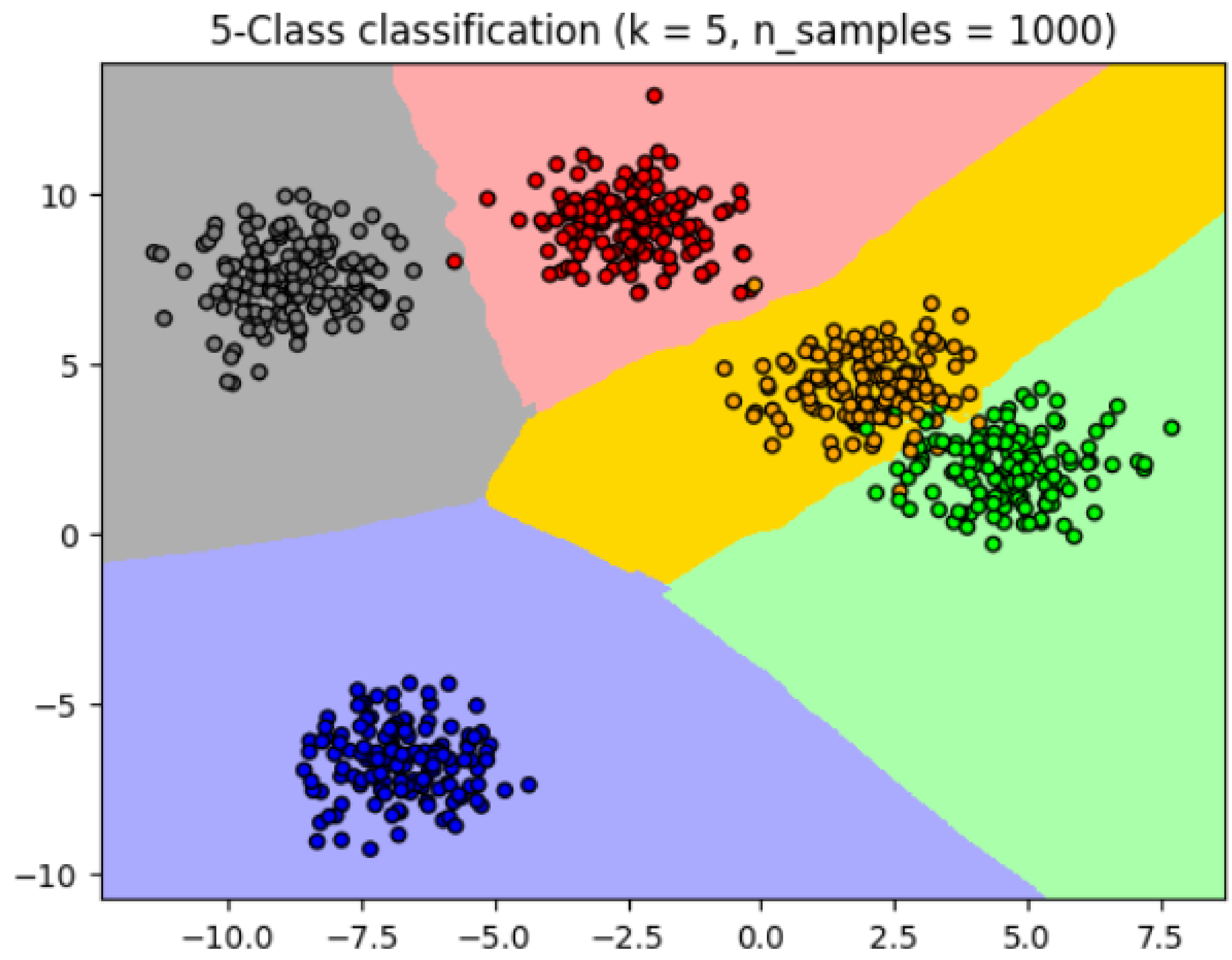

Figure 3.9: A decision boundary plot created by ChatGPT Plus, GPT-4

We can draw insights from this experiment:

- Natural language modeling trains a model to predict words or sequences by considering the context.
- **Natural Language Understanding** (**NLU**) takes the model into the depths of the intents and meanings of human language.

We could go on like this, having conversations with ChatGPT Plus GPT-4 and producing an unlimited amount of documented and commented source code in a variety of languages.We are now ready to tackle downstream tasks.

Investigating the potential of downstream tasks

Transformers, like humans, can be fine-tuned to perform downstream tasks by inheriting the properties of a pretrained model. The pretrained model provides its architecture and language representations through its parameters.A pretrained model trains on key tasks to acquire a general knowledge of the language. A fine-tuned model trains on downstream tasks. Not every transformer model uses the same tasks for pretraining. But, potentially, all tasks can be pretrained or fine-tuned.Organizing downstream tasks provides a scientific framework for implementing and measuring NLP. However, every NLP model needs to be evaluated with a standard method.This section will first go through some of the key measurement methods. Then, we will go through some of the main benchmark tasks and datasets.Let's start by going through some of the key metric methods.

Evaluating models with metrics

It is impossible to compare one transformer model to another (or any other NLP model) without a universal measurement system that uses metrics. In this section, we will analyze some of the measurement scoring methods.

Accuracy score

In whatever variant you use, the accuracy score is a practical evaluation. The score function calculates an exact true or false value for each result. Either the model's outputs, $\hat{y}$, match the correct predictions, y, for a given subset, *samples*i, of a set of samples or not. The primary function is:

$$Accuracy(y, \hat{y}) = \frac{1}{n_{samples}} \sum_{i=0}^{n_{samples}-1} 1(\hat{y}_i = y_i)$$

Copy Explain

```
We will obtain 1 if the result for the subset is correct and 0 if it is false.
```

Let's now examine the more flexible F1-score.

F1-score

The F1-score introduces a more flexible approach that can help when faced with datasets containing uneven class distributions.In this equation, true (*T*) positives (*p*), false (*F*) positives (*p*), and false (*F*) negatives (*n*) are first plugged into the precision (*P*) and recall (*R*) equations :

$$P = \frac{T_p}{T_p + F_p}$$

Then the F1-score uses precision(*P*) and recall(*R*) to build the metric: *F1score= 2* (precision * recall)/(precision + recall)*The F1-score can thus be viewed as the harmonic mean (reciprocal of the arithmetic mean) of precision (*P*) and recall (*R*):

$$F1\ score = 2 \times \frac{P \times R}{P + R}$$

Let's now review the MCC approach.

Matthews Correlation Coefficient (MCC)

The Accuracy score and F1-score are ways to measure the the output of a transformer model objectively. The Matthews Correlation Coefficient (MCC) provides another measurement tool. These tools help us compare the same model with different configurations. They also help us rank transformer models.MCC will be implemented in the *Evaluating using Matthews Correlation Coefficient* section in *Chapter 5, Diving into Fine-Tuning through BERT*. MCC computes a measurement with true positives (*T*P), true negatives (*T*N), false positives (*F*P), and false negatives (*F*N).The MCC can be summarized by the following equation:

$$\frac{TP\ x\ TN - FP\ x\ FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

MCC provides an excellent metric for binary classification models, even if the sizes of the classes are different.We now have a good idea of how to measure a given transformer model's results and compare them to other transformer models or NLP models.With some of the scoring methods in mind, we will now see how human evaluation can contribute to valuable measurements.

Human Evaluation

Human evaluations can apply to smaller datasets or subsets of larger datasets.Human evaluations can be effective to:

- analyze the outputs of a model during the design and training process
- provide feedback once the model is in production
- create datasets of samples for a given task
- create new benchmark tasks and methods

With measurement scoring methods in mind, let's look into benchmark tasks and datasets.

Benchmark tasks and datasets

Three prerequisites are required to prove that transformers have reached state-of-the-art performance levels:

A model

A dataset-driven task

A metric as described in the *Evaluating models with metrics* section of this chapterThere are many benchmarks on the market, such as Google's Big Benchmark, https://github.com/google/BIG-bench, that we will use to explore self-evaluations in *Chapter 15, Guarding the Giants: Mitigating Risks in Large Language Models*, in *The Emergence of Functional AGI* section.*However, it makes no sense to implement all the benchmarks. It is essential to run enough tasks to understand the potential of Natural Language Processing tasks with transformers. From there, you will quickly adapt to any benchmarking system.* We will begin by exploring the SuperGLUE benchmark to illustrate the evaluation process of a transformer model through some of the main NLP tasks.

From GLUE to SuperGLUE

Many Natural Language Processing benchmarks can be applied to measure a model's performance, such as GLUE, SuperGLUE, and WMT(Workshop on Machine Translation. We will use WMT in *Chapter 4, Advancements in Translations with Google Trax, Google Translate, and Google Bard*. This chapter focuses on the SuperGLUE benchmark. The SuperGLUE benchmark was designed and made public by *Wang* et al. (2019). *Wang* et al. (2019) first designed the **General Language Understanding Evaluation** (**GLUE**) benchmark.The motivation of the GLUE benchmark was to show that to be useful, NLU has to be applicable to a wide range of tasks. Relatively small GLUE datasets were designed to encourage an NLU model to solve a set of tasks.However, the performance of NLU models, boosted by the arrival of transformers, began to exceed the average level. The GLUE leaderboard, available at https://gluebenchmark.com/leaderboard, shows a remarkable display of NLU talent, mainly concentrating on the ground-breaking transformer models. The leaderboard makes less sense than understanding and applying the benchmark tasks to a model to see if it fits our needs. We can enhance the datasets for specific projects. Human baselines are far down the list:

21	Facebook AI	RoBERTa
22	Microsoft D365 AI & MSR AI	MT-DNN-ensemble
23	GLUE Human Baselines	GLUE Human Baselines

Figure 3.10: GLUE Leaderboard November 2023

New models and the Human Baselines ranking will constantly change. The position of Human Baselines doesn't mean much anymore since the arrival of foundation models. These rankings just show how far classical NLP and transformers have taken us!GLUE Human Baselines are not in a top position, which shows that NLU models have surpassed non-expert humans on GLUE tasks. Human Baselines represent what we humans can achieve. AI can now outperform humans. However, it is challenging to blindly fish around for benchmark datasets to improve our models without a standard to refer to.We also notice that transformer models have taken the lead in the 2020s. This trend will continue, although we never know what creative innovators will find!I like to think of GLUE and SuperGLUE as the point when words go from chaos to order with language understanding. Understanding is the "glue" that makes words fit together and become a language.The GLUE leaderboard will continuously evolve as NLU progresses. However, *Wang* et al. (2019) introduced SuperGLUE to set a higher standard for Human Baselines.

Introducing higher Human Baselines standards

Wang et al. (2019) recognized the limits of GLUE. They designed SuperGLUE for more difficult NLU tasks.SuperGLUE helped improve the position of Human Baselines as shown in the following excerpt of the leaderboard: https://super.gluebenchmark.com/leaderboard:

6	Zirui Wang	T5 + UDG, Single Model (Google Brain)
7	DeBERTa Team - Microsoft	DeBERTa / TuringNLRv4
8	SuperGLUE Human Baselines	SuperGLUE Human Baselines

Figure 3.11: SuperGLUE Leaderboard 2.0 – November 2023

The SuperGLUE leaderboard keeps evolving, and AI will continue to push human baselines of basic NLP tasks down. The top models are not displayed in *Figure 3.5* for two reasons:

- The figure does not include some of the top proprietary foundation models.

The ranking changes constantly with models specialized in SuperGlue tasks, which are an excellent first-level assessment of a model. However, you will need to evaluate models according to your needs to make a choice.

So yes, AI algorithm rankings will constantly change as new innovative models arrive. However, these rankings just show how hard the battle for NLP supremacy is being fought!The SuperGLUE evaluation process is more important than the rankings that will keep changing.

Note: The best model for you is not necessarily the best model on a leaderboard. You need to learn an NLP evaluation process and apply it to the model(s) you choose to implement.

Let's now see how the evaluation process works.

The SuperGLUE evaluation process

Wang et al. (2019) selected practical representative tasks of NLP for their SuperGLUE benchmark. The selection criteria for these tasks were stricter than for GLUE. For example, the tasks had to not only understand texts but also reason. The level of reasoning is not that of a top human expert. However, the level of performance is sufficient to replace many human tasks.The main SuperGLUE tasks are presented in a ready-to-use list:

SuperGLUE Tasks

Name	Identifier	Download	More Info	Metric
Broadcoverage Diagnostics	AX-b			Matthew's Corr
CommitmentBank	CB			Avg. F1 / Accuracy
Choice of Plausible Alternatives	COPA			Accuracy
Multi-Sentence Reading Comprehension	MultiRC			F1a / EM
Recognaing Textual Entailment	RTE			Accuracy
Words in Context	WiC			Accuracy
The Winograd Schema Challenge	WSC			Accuracy
BoolQ	BoolQ			Accuracy
Reading Comprehension with Commonsense Reasoning	ReCoRD			F1 / Accuracy
Winogender Schema Diagnostics	AX-g			Gender Parity / Accuracy

DOWNLOAD ALL DATA

Figure 3.12: SuperGLUE tasks

The task list is interactive: https://super.gluebenchmark.com/tasks.Each task contains links to the required information to perform that task:

Name is the name of the downstream task of a fine-tuned, pretrained model

Identifier is the abbreviation or short version of the name

Download is the download link to the datasets

More Info offers greater detail through a link to the paper or website of the team that designed the dataset-driven task(s)

Metric is the measurement score used to evaluate the model

SuperGLUE provides the task instructions, the software, the datasets, and papers or websites describing the problem to be solved. Once a team runs the benchmark tasks and reaches the leaderboard, the results are displayed:

Score	BoolQ	CB	COPA	MultiRC	ReCoRD	RTE	WiC	WSC	AX-b	AX-g
89.8	89.0	95.8/98.9	100.0	81.8/51.9	91.7/91.3	93.6	80.0	100.0	76.6	99.3/99.7

Figure 3.13: SuperGLUE task scores

SuperGLUE displays the overall score and the score for each task.For example, let's take the instructions *Wang* et al. (2019) provided for the **Choice of Plausible Answers (COPA)** task in *Table 6* of their paper.The first step is to read the remarkable paper by *Roemmele* et al. (2011). In a nutshell, the goal is for the NLU model to demonstrate its machine thinking (not human thinking, of course) potential. In our case, the Transformer must choose the most plausible answer to a question. The dataset provides a premise, and the transformer model must find the most plausible answer.For example:

Copy Explain

```
Premise: I knocked on my neighbor's door.
What happened as a result?
Alternative 1: My neighbor invited me in.
Alternative 2: My neighbor left his house.
This question requires a second or two for a human to answer, which shows that it
requires some common sense machine thinking. COPA.zip, a ready-to-use dataset, can be
downloaded directly from the SuperGLUE task page. The metric provided makes the
process equal and reliable for all participants in the benchmark race.
```

COPA testing opens the door to advanced question-and-answer tasks, as we will see in Chapter 12.We have introduced COPA. Let's define some of the other SuperGLUE benchmark tasks.

Defining the SuperGLUE benchmark tasks

A task can be a pretraining task to generate a trained model. That same task can be a downstream task for another model that will fine-tune it. However, the goal of SuperGLUE is to show that a given NLU model can perform multiple downstream tasks with fine-tuning. Multi-task models are the ones that prove the thinking power of transformers.The power of any transformer resides in its ability to perform multiple tasks using a pretrained model and then applying it to fine-tuned downstream tasks. The original Transformer model and its variants are now widely present in the top rankings for all the GLUE and SuperGLUE tasks. We will continue to focus on SuperGLUE downstream tasks for which Human Baselines are tough to beat.In the previous section, we went through COPA. In this section, we will go through seven other tasks defined by *Wang* et al. (2019) in *Table 2* of their paper.Let's continue with a Boolean question task.

BoolQ

BoolQ is a Boolean yes-or-no answer task. The dataset, as defined on SuperGLUE, contains 15,942 naturally occurring examples. A raw sample of line #3 of the train.jsonl dataset contains a passage, a question, and the answer (true):

Copy Explain

```
{"question": "is windows movie maker part of windows essentials"
"passage": "Windows Movie Maker -- Windows Movie Maker (formerly known as Windows Live
Movie Maker in Windows 7) is a discontinued video editing software by Microsoft. It is
a part of Windows Essentials software suite and offers the ability to create and edit
videos as well as to publish them on OneDrive, Facebook, Vimeo, YouTube, and Flickr.",
"idx": 2, "label": true}
```

The datasets provided may change in time, but the concepts remain the same.Now, let's examine CB, a task that requires both humans and machines to focus.

Commitment Bank (CB)

Commitment Bank (CB) is a difficult *entailment* task. We are asking the transformer model to read a *premise* and then examine a *hypothesis* built on the premise. For example, the hypothesis will confirm the premise or contradict it. Then, the transformer model must *label* the hypothesis as *neutral*, an *entailment*, or a *contradiction* of the premise, for example.The dataset contains natural discourses.

Copy Explain

```
The following sample , #77, taken from the train.jsonl training dataset shows how
difficult the CB task is:
{"premise": "The Susweca. It means ''dragonfly'' in Sioux, you know. Did I ever tell
you that's where Paul and I met?"
"hypothesis": "Susweca is where she and Paul met,"
"label": "entailment", "idx": 77}
```

We will now have a look at the multi-sentence problem.

Multi-Sentence Reading Comprehension (MultiRC)

Copy Explain

```
Multi-Sentence Reading Comprehension (MultiRC) asks the model to read a text and
choose from several possible choices. The task is difficult for both humans and
machines. The model is presented with a text, several questions, and possible answers
to each question with a 0 (false) or 1 (true) label.
Let's take the second sample in train.jsonl:
"Text": "text": "The rally took place on October 17, the shooting on February 29.
Again, standard filmmaking techniques are interpreted as smooth distortion: \"Moore
works by depriving you of context and guiding your mind to fill the vacuum -- with
completely false ideas. It is brilliantly, if unethically, done.\" As noted above, the
\"from my cold dead hands\" part is simply Moore's way to introduce Heston. Did anyone
but Moore's critics view it as anything else? He certainly does not \"attribute it to
a speech where it was not uttered\" and, as noted above, doing so twice would make no
sense whatsoever if Moore was the mastermind deceiver that his critics claim he is.
Concerning the Georgetown Hoya interview where Heston was asked about Rolland, you
write: \"There is no indication that [Heston] recognized Kayla Rolland's case.\" This
is naive to the extreme -- Heston would not be president of the NRA if he was not kept
up to date on the most prominent cases of gun violence. Even if he did not respond to
that part of the interview, he certainly knew about the case at that point. Regarding
the NRA website excerpt about the case and the highlighting of the phrase \"48 hours
after Kayla Rolland is pronounced dead\": This is one valid criticism, but far from
the deliberate distortion you make it out to be; rather, it is an example for how the
facts can sometimes be easy to miss with Moore's fast pace editing. The reason the
sentence is highlighted is not to deceive the viewer into believing that Heston
hurried to Flint to immediately hold a rally there (as will become quite obvious), but
simply to highlight the first mention of the name \"Kayla Rolland\" in the text, which
is in this paragraph. "
```

The sample contains four questions. To illustrate the task, we will investigate two of them. The model must predict the correct labels. Notice how the information that the model is asked to obtain is distributed throughout the text:

Copy Explain


```
"question": "When was Kayla Rolland shot?"
"answers":
[{"text": "February 17", "idx": 168, "label": 0},
{"text": "February 29", "idx": 169, "label": 1},
{"text": "October 29", "idx": 170, "label": 0},
{"text": "October 17", "idx": 171, "label": 0},
{"text": "February 17", "idx": 172, "label": 0}], "idx": 26},
{"question": "Who was president of the NRA on February 29?",
"answers": [{"text": "Charleton Heston", "idx": 173, "label": 1},
{"text": "Moore", "idx": 174, "label": 0},
{"text": "George Hoya", "idx": 175, "label": 0},
{"text": "Rolland", "idx": 176, "label": 0},
{"text": "Hoya", "idx": 177, "label": 0}, {"text": "Kayla", "idx": 178, "label": 0}],
"idx": 27},
```

At this point, one can only admire the performance of a fine-tuned, pretrained model on these difficult downstream tasks.Now, let's see the reading comprehension task.
Reading Comprehension with Commonsense Reasoning Dataset (ReCoRD)

Reading Comprehension with Commonsense Reasoning Dataset (ReCoRD) represents another challenging task. The dataset contains over 120,000 queries from more than 70,000 news articles. The Transformer must use commonsense reasoning to solve this problem.

Copy Explain

```
Let's examine a sample from train.jsonl:
"source": "Daily mail"
A passage contains the text and indications as to where the entities are located.
A passage begins with the text:
"passage": {
    "text": "A Peruvian tribe once revered by the Inca's for their fierce hunting
skills and formidable warriors are clinging on to their traditional existence in the
coca growing valleys of South America, sharing their land with drug traffickers,
rebels and illegal loggers. Ashaninka Indians are the largest group of indigenous
people in the mountainous nation's Amazon region, but their settlements are so sparse
that they now make up less than one per cent of Peru's 30 million population. Ever
since they battled rival tribes for territory and food during native rule in the
rainforests of South America, the Ashaninka have rarely known peace.\n@highlight\nThe
Ashaninka tribe once shared the Amazon with the like of the Incas hundreds of years
ago\n@highlight\nThey have been forced to share their land after years of conflict
forced rebels and drug dealers into the forest\n@highlight\n. Despite settling in
valleys rich with valuable coca, they live a poor pre-industrial existence",
```

The *entities* are indicated, as shown in the following excerpt:

Copy Explain

```
  "entities": [{"start": 2,"end": 9}, …,"start": 711,"end": 715}]
```

Finally, the model must *answer* a *query* by finding the proper value for the *placeholder*:

Copy Explain

```
{"query": "Innocence of youth: Many of the @placeholder's younger generations have
turned their backs on tribal life and moved to the cities where living conditions are
better",
"answers":[{"start":263,"end":271,"text":"Ashaninka"},
{"start":601,"end":609,"text":"Ashaninka"},
{"start":651,"end":659,"text":"Ashaninka"}],"idx":9}],"idx":3}
```

Once the transformer model has gone through this problem, it must now face an entailment task.

Recognizing Textual Entailment (RTE)

For **Recognizing Textual Entailment (RTE)**, the transformer model must read the *premise*, examine a *hypothesis*, and predict the *label* of the *entailment hypothesis status*.

Copy Explain

```
Let's examine sample #19 of the train.jsonl dataset:
{"premise": "U.S. crude settled $1.32 lower at $42.83 a barrel.",
"hypothesis": "Crude the light American lowered to the closing 1.32 dollars, to 42.83
dollars the barrel.", "label": "not_entailment", >"idx": 19}
```

RTE requires understanding and logic. Let's now see the Words in Context task.

Words in Context (WiC)

Words in Context (WiC) and the following Winograd task test a model's ability to process an ambiguous word. In WiC, the multi-task Transformer will have to analyze two sentences and determine whether the target word has the same meaning in both sentences.

Copy Explain

```
Let's examine the first sample of the train.jsonl dataset.
```

First, the target word is specified:

Copy Explain

```
  "word": "place"
```

The model has to read two sentences containing the target word:

Copy Explain

```
  "sentence1": "Do you want to come over to my place later?",
  "sentence2": "A political system with no place for the less prominent groups."
train.jsonl specifies the sample index, the value of the label, and the position of
the target word in sentence1(start1, end1) and sentence2(start2, end2):
  "idx": 0,
  "label": false,
  "start1": 31,
  "start2": 27,
  "end1": 36,
  "end2": 32,
```

After this daunting task, the transformer model has to face the Winograd task.

The Winograd Schema Challenge (WSC)

The Winograd schema task is named after Terry Winograd. If a transformer is well-trained, it should be able to solve disambiguation problems.The dataset contains sentences that target slight differences in the gender of a pronoun.This constitutes a coreference resolution problem, which is one of the most challenging tasks to perform. However, the transformer architecture, which allows self-attention, is ideal for this task.Each sentence contains an *occupation*, a *participant*, and a *pronoun*. The problem is to find whether the pronoun is *coreferent* with the occupation or the participant.

Copy Explain

```
Let's examine a sample taken from train.jsonl.
```

First, the sample asks the model to read a *text*:

Copy Explain

```
{"text": >"I poured water from the bottle into the cup until it was full.",
The WSC ask the model to find the target pronoun token number 10 starting at 0:
"target": {"span2_index": 10,
```

Then, it asks the model to determine if "it" refers to "the cup" or not:

Copy Explain

```
"span1_index": 7,
"span1_text": "the cup",
"span2_text": "it"},
For sample index #4, the label is true:
"idx": 4, "label": true}
```

We have gone through some of the main SuperGLUE tasks. There are many other tasks.However, once you understand the transformers' architecture and the benchmark tasks' mechanism, you will rapidly adapt to any model and benchmark.Let's now run some downstream tasks.

Running downstream tasks

In this section, we will jump into some transformer cars and drive them around a bit to see what they do. There are many models and tasks. We will run a few of them in this section. We will be going through variants of these models during our journey in the book. Once you understand the process of running a few tasks, you will quickly understand all of them. *After all, the human baseline for all these tasks is us!*A downstream task is a fine-tuned transformer task that inherits the model and parameters from a pretrained transformer model.A downstream task is thus the perspective of a pretrained model running fine-tuned tasks. That means, depending on the model, a task is downstream if it was not used to fully pretrain the model. In this section, we will consider all the tasks as downstream since we did not pretrain them.Models will evolve, as will databases, benchmark methods, accuracy measurement methods, and leaderboard criteria. However, the structure of human thought reflected through the downstream tasks in this chapter will remain.Let's start with CoLA.

The Corpus of Linguistic Acceptability (CoLA)

The **Corpus of Linguistic Acceptability (CoLA)**, a GLUE task, https://gluebenchmark.com/tasks, contains thousands of samples of English sentences annotated for grammatical acceptability.The goal of *Alex Warstadt* et al. (2019) was to evaluate the linguistic competence of an NLP model to judge the linguistic acceptability of a sentence. The NLP model is expected to classify the sentences accordingly.

Copy Explain

```
The sentences are labeled as grammatical or ungrammatical. The sentence is labeled 0
if the sentence is not grammatically acceptable. The sentence is labeled 1 if the
sentence is grammatically acceptable. For example:
Classification = 1 for 'we yelled ourselves hoarse.'
Classification = 0 for 'we yelled ourselves.'
```

You can go through BERT_Fine_Tuning_Sentence_Classification_GPU.ipynb in *Chapter 5, Diving into Fine-Tuning through BERT*, to view the BERT model that we fine-tuned on CoLA datasets. We used CoLA data.We will first load the dataset:

Copy Explain

```
#source of dataset : https://nyu-mll.github.io/CoLA/
df = pd.read_csv("in_domain_train.tsv", delimiter='\t', header=None, names=
['sentence_source', 'label', 'label_notes', 'sentence'])
df.shape
```

We will also load a pretrained Hugging Face BERT model:

Copy Explain

```
model = BertForSequenceClassification.from_pretrained("bert-base-uncased",
num_labels=2)
```

Finally, the measurement method, or metric, we used is MCC, which was described in the *Matthews Correlation Coefficient (MCC)* section of this chapter.You can refer to that section for the mathematical description of MCC and take the time to rerun the source code if necessary.A sentence can be grammatically unacceptable but still convey a sentiment. Sentiment analysis can add some form of empathy to a machine.

Stanford Sentiment TreeBank (SST-2)

Stanford Sentiment TreeBank (**SST-2**) contains movie reviews. This section will describe the SST-2 (binary classification) task. However, the datasets go beyond that, and it is possible to classify sentiments in a range of *0* (negative) to *n* (positive).*Socher* et al. (2013) took sentiment analysis beyond the binary positive-negative NLP classification. We will explore the SST-2 multi-label sentiment classification with a transformer model in *Chapter 14, Advanced Sentiment Analysis*.In this section, we will run a sample taken from SST on a Hugging Face transformer pipeline model to illustrate binary classification.

Copy Explain

```
Open Transformer_tasks.ipynb and run the following cell, which contains positive and
negative movie reviews taken from SST-2 for binary classification:
from transformers import pipeline
nlp = pipeline("sentiment-analysis", model="distilbert-base-uncased-finetuned-sst-2-
english")
print(nlp("If you sometimes like to go to the movies to have fun , Wasabi is a good
place to start .")),"If you sometimes like to go to the movies to have fun , Wasabi is
a good place to start .")
print(nlp("Effective but too-tepid biopic.")),"Effective but too-tepid biopic.")
```

The output is accurate:

Copy Explain

```
[{'label': 'POSITIVE', 'score': 0.9998257756233215}] If you sometimes like to go to
the movies to have fun , Wasabi is a good place to start .
[{'label': 'NEGATIVE', 'score': 0.9974064230918884}] Effective but too-tepid biopic.
```

The SST-2 task is evaluated using the accuracy metric.We classify sentiments of a sequence. Let's now see whether two sentences in a sequence are paraphrases or not.

Microsoft Research Paraphrase Corpus (MRPC)

The **Microsoft Research Paraphrase Corpus (MRPC)**, a GLUE task, contains pairs of sentences extracted from new sources on the web. A human has annotated each pair to indicate whether the sentences are equivalent based on two closely related properties:

- Paraphrase equivalent
- Semantic equivalent

Copy Explain

```
Let's run a sample using the Hugging Face BERT model. Open Transformer_tasks.ipynb,
go to the following cell and then run the sample taken from MRPC:
from transformers import AutoTokenizer, TFAutoModelForSequenceClassification
import tensorflow as tf
tokenizer = AutoTokenizer.from_pretrained("bert-base-cased-finetuned-mrpc")
model = TFAutoModelForSequenceClassification.from_pretrained("bert-base-cased-
finetuned-mrpc")
classes = ["not paraphrase", "is paraphrase"]
sequence_A = "The DVD-CCA then appealed to the state Supreme Court."
sequence_B = "The DVD CCA appealed that decision to the U.S. Supreme Court."
paraphrase = tokenizer.encode_plus(sequence_A, sequence_B, return_tensors="tf")
paraphrase_classification_logits = model(paraphrase)[0]
paraphrase_results = tf.nn.softmax(paraphrase_classification_logits, axis=1).numpy()[0]
print(sequence_B, "should be a paraphrase")
for i in range(len(classes)):
    print(f"{classes[i]}: {round(paraphrase_results[i] * 100)}%")
```

The output is accurate:

Copy Explain

```
The DVD CCA appealed that decision to the U.S. Supreme Court. should be a paraphrase
not paraphrase: 8%
is paraphrase: 92%
```

The MRPC task is measured with the F1/Accuracy score method.Let's now run a Winograd schema.

Winograd schemas

We described the Winograd schemas in this Chapter's *The Winograd Schema Challenge (WSC)* section. The training set was in English.But what happens if we ask a transformer model to solve a pronoun gender problem in an English-French translation? French has different spellings for nouns that have grammatical genders (feminine, masculine).The following sentence contains the pronoun *it*, which can refer to the words *car* or *garage*. Can a transformer disambiguate this pronoun?

Copy Explain

```
Open Transformer_tasks.ipynb, go to the #Winograd cell, and run our example:
from transformers import pipeline
translator = pipeline("translation_en_to_fr",model="t5-base")
print(translator("The car could not go in the garage because it was too big.",
max_length=40))
```

The translation is acceptable:

Copy Explain

```
[{'translation_text': "La voiture ne pouvait pas aller dans le garage parce qu'elle
était trop grosse."}]
```

The transformer detected that the word *it* refers to the word *car*, which is a feminine form. The feminine form applies to *it*, and the adjective *big*:*elle* means *she* in French, which is the translation of *it*. The masculine form would have been *il*, which means *he*.*grosse* is the feminine form of the translation of the word *big*. Otherwise, the masculine form would have been *gros*.We gave the transformer a difficult Winograd schema to solve, and it produced the correct answer.There are many more dataset-driven NLU tasks available. We will explore some of them throughout this book to add more building blocks to our toolbox of transformers.

Summary

The paradigm shift triggered by ChatGPT compelled us to redefine what an NLP task is. We saw that ChatGPT, like other LLM models, can perform tasks they were not trained for, including many SuperGLUE tasks through advanced emergence. We explored the outputs of the attention heads to bridge the gap between numerical calculations and producing sequences of words.We then explored how to measure the performance of multi-task transformers. Transformers' ability to obtain top-ranking results for downstream tasks is unique in NLP history. We went through the demanding SuperGLUE tasks that brought transformers up to the top ranks of the GLUE and SuperGLUE leaderboards.BoolQ, CB, WiC, and the many other tasks we covered are by no means easy to process, even for humans. We went through an example of several downstream tasks that show the difficulty transformer models face in proving their efficiency.Transformers have proven their value by outperforming the former NLU architectures. To illustrate how simple it is to implement downstream fine-tuned tasks, we then ran several tasks in a Google Colaboratory notebook using Hugging Face's pipeline for transformers.In *Winograd schemas*, we gave the Transformer the difficult task of solving a Winograd disambiguation problem for an English-French translation.In this chapter, we went through the NLP benchmarking process with SuperGLUE and some tasks. For more, consult the Further Reading section of this chapter. *Chapter 4*, *Advancements in Translations with Google Trax, Google Translate, and Google Bard*, will take translation tasks a step further, and we will build a translation model with Trax.

Questions

1. Machine intelligence uses the same data as humans to make predictions. (True/False)
2. SuperGLUE is more difficult than GLUE for NLP models. (True/False)
3. BoolQ expects a binary answer. (True/False)
4. WiC stands for Words in Context. (True/False)

5. **Recognizing Textual Entailment** (**RTE**) detects whether one sequence entails another sequence. (True/False)
6. A Winograd schema predicts whether a verb is spelled correctly. (True/False)
7. Transformer models now occupy the top ranks of GLUE and SuperGLUE. (True/False)
8. Human Baselines standards are not defined once and for all. They were made tougher to attain by SuperGLUE. (True/False)
9. Transformer models will never beat SuperGLUE Human Baselines standards. (True/False)
10. Variants of transformer models have outperformed RNN and CNN models. (True/False)

References

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, Samuel R. Bowman, 2019, *SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems*: https://w4ngatang.github.io/static/papers/superglue.pdf

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, Samuel R. Bowman, 2019, *GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding*: https://arxiv.org/abs/1804.07461

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, Haifeng Wang, 2019, *ERNIE 2.0: A Continual Pretraining Framework for Language Understanding*: https://arxiv.org/pdf/1907.12412.pdf

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon, 2011, *Choice of Plausible Alternatives: An Evaluation of Commonsense Causal Reasoning*: https://people.ict.usc.edu/~gordon/publications/AAAI-SPRING11A.PDF

Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts, 2013, *Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank*: https://nlp.stanford.edu/~socherr/EMNLP2013_RNTN.pdf

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Jamie Brew, 2019, *HuggingFace's Transformers: State-of-the-art Natural Language Processing*: https://arxiv.org/abs/1910.03771

Hugging Face transformer Usage:
https://huggingface.co/transformers/usage.html

Further Reading

You can examine many other large language model benchmarking approaches, including the following tools:

- Hugging Face is a good place to continue with the Open LLM Leaderboard : https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard
- Google provides a comprehensive benchmarking service with BIG-bench and 200+ tasks: https://github.com/google/BIG-bench

Ultimately, the decision to use a benchmarking framework depends on each project.