Preliminary Survey on Foundation Language Models

Vyom Pathak
Department of Computer & Information Science & Engineering
University of Florida
Gainesville, FL, USA
v.pathak@ufl.edu

***Abstract*—Language models are essential components of natural language processing that can capture general representations of language from large-scale text corpora. In recent years, various language models have been proposed, ranging from shallow word embeddings to deep contextual encoders, with different pre-training tasks, training frameworks, adaptation methods, and evaluation benchmarks. The year 2022 saw a surge in the development of large generative models, referred to as "foundation models," that can perform multiple tasks by training on a general unlabeled dataset. However, there is a need for a comprehensive survey that can link these models and contrast their advantages and disadvantages. In this paper, we present a systematic survey of foundation language models that aims to connect various models based on multiple criteria, such as representation learning, model size, task capabilities, research questions, and practical task capabilities. We also conduct experiments on a subset of SuperGLUE tasks using six representative models from different architecture families and contrast their performance and efficiency. Moreover, we suggest some future directions that can be pursued to improve the robustness and comprehensiveness of this survey. We release code for mode training, and evaluation, the paper draft, slides, and video talk here: https://github.com/01-vyom/Evolving-Survey-on-Foundational-Language-Models**

***Index Terms*—foundation models, large language models, natural language processing, pre-training frameworks, language representation learning**

I. Introduction

With the advent of deep learning, various neural network-based models have taken over the realm of Natural Language Processing (NLP). These include convolutional neural networks [38], recurrent neural networks [93], graph-based neural networks [90], attention based models [4, 97]. One of the major advantages of these models is their ability to palliate feature engineering problems [93].

Statistical NLP models depend on engineered features, while neural language models use dense representations to capture the semantic, syntactic, and contextual features of the text [70]. As these features are learned in-task, it becomes easy for the general public to develop various large-scale natural language systems.

Contrasting the depth of the neural networks, deep NN's tend to have a huge amount of parameters resulting in overfitted results for low-resource datasets with a little generalization ability [70]. This resulted in shallow language models in the earlier periods for various NLP tasks [64].

Lately, there has been an uptick in leveraging different frameworks for learning general representations to mitigate these problems [70]. One of the most common methods amongst these is pre-training language models (PTLM) on large text corpus leveraging universal language representation. These representations are then used for downstream NLP tasks showing great performance boosts as compared to training models from scratch [22]. This combined with the development of processing power (GPU), there has been an emergence of deep models [22]. Hand-in-hand the research community saw constant improvement in model development (from training to inference), and architectural improvements. This has led to deep models being used in day-to-day applications instead of shallow models.

Pre-cursors to the modern pre-training models, the models learned word embeddings. This constituted shallow models as the model itself is not needed for downstream tasks [55, 63]. Majorly, these models captured the semantic, and syntactic representation of language but lacked the contextual high-level concepts of the sentence. The next generation of models learned contextual word embeddings [22].

This lead to models which are trained with a large amount of data at an immense model parameters scale, and being adapted (fine-tuning [51], in-context learning [25] to a broad category of downstream tasks. These types of models are formally known as foundation models or large language models (LLMs) [48].

We present a survey on language models which aims at connecting each model to each other based on the following criteria:

- Representation learning: As language modeling constitutes learning statistical patterns, semantic, and contextual meaning, models differ based on how they are developed for each criterion. We show which types of representation are learned from each model family [7].
- Model Size: With the growth of computing availability, there has been a surge of LLMs from several different organizations, and institutes. The main-stream research shows the largest models of a particular model family, but we see that there could be multiple models in a family (e.g. BERT-base, BERT-large) [22]. So, we show a family of LLMs with different model sizes.

- Task Capabilities: Given an LLM's vast amount of size, we need to show what is the task capability of each model. This has been a continuous trend in the NLP community going from individual datasets [75] to small collections of datasets [98], to large collections of datasets [11, 91]. However, it is impossible to consider all the task scenarios a model can perform. So, to tackle these questions we link models based on their task-completion abilities.
- Research Questions: Each model plans to tackle different research questions building on top of each other. This ranges from training schemes, architectural innovation, model limitations, availability, in-context learning, social impact, ...etc. This becomes the most significant way to connect and show different models across the time span.
- Practical Task Capabilities: To perform a task completion comparison on a practical basis, we choose multiple common NLP tasks, and performance evaluation of select models (based on an architectural criterion, and certain training schemes)[1].

The rest of the survey is organized as follows: Section II outlines the definitions, and concepts used for differentiating the language models. Section III goes over different models with descriptions of each model in chronological order, discussing different criteria defined above. Section IV goes over the experimentation setup including dataset details, training detail, evaluation detail for a few selected models, and future work. Section V goes over the conclusion of the proposed work.

II. Background

A. Language Representation Learning

For language, representations must be good enough to capture the linguistic rules, such as syntactic structures, lexical meanings, and semantic roles. The fundamental concept behind language representation learning is to represent the meaning of a text using low-dimensional real-valued vectors. In general, there are two types of embeddings: non-contextual, and contextual embeddings. Non-contextual is a static representation of the training data, while contextual representation changes dynamically according to the context of the sentence [70].

For non-contextual embeddings, the language symbols are directly mapped into a distributed embedding space. These embeddings are trained on task data with other model parameters. These embeddings have limitations such that they are static in nature i.e. once they are calculated, they are used as is for each word even if the context of the word changes, and because of this, they cannot handle out-of-vocabulary words very well. This led to the development of character-level as well as sub-word-level representation [10, 39, 87].

To mitigate all these issues, contextual embeddings [64] were introduced which capture the context of the word. More broadly, this constitutes neural contextual encoders which can further be classified into sequential, and non-sequential models.

Sequential models aim at capturing the local context of a word in sequential order [70]. Usually, this involves using convolutional models like CNN, and recurrent models like LSTMs, GRUs, and bidirectional-LSTMs [12, 30, 38, 86].

Non-sequential models learn contextual representation using a tree or a graph with context-aware structure between words, capturing syntactic, structures or semantic relations [40, 90, 94, 107]. Inspired by the graphical representation, the modern development of the self-attention mechanism as a fully-connected graph that learns the connection weights dynamically has been a revolution to the field of NLP [97]. Because of the non-sequential nature of this algorithm, the sequential information is learned using positional embeddings, position-wise multi-layer perception, and residual connections.

B. Training Frameworks

With the advent of large language models, large amounts of datasets are needed to train the model as well as avoid overfitting fully. This becomes a bottleneck in terms of resources to develop large-scale labeled datasets. On the contrary, unlabeled text corpora on large scale are easy to construct because of the amount of data generated every day on the internet. This leads to the pre-training of models on large unannotated corpora, which are then used for downstream tasks [22]. This not only provides better model initialization, leading to better generalization performance, and speedier convergence but also acts as a regularization measure to avoid overfitting on small data [26].

Pre-training to learn represented words as dense vectors are generally divided into 2 categories: the first is pre-trained word embeddings, and the second is pre-trained contextual encoders. The first ever word embeddings were introduced in neural network language models (NNLM) [8]. Followed by that, to address the computational complexity of training on large unlabelled data, Collobert et al. [16] using the pairwise ranking task. Mokolov et al. presented that shallow models can also build decent word embeddings [56] by proposing the pioneered architectures: Continuous Bag-of-Words, and Skip-Gram models. Despite their plainness, these models can learn high-quality word embeddings to capture semantic, and syntactic similarities among words. To capture global patterns, Glove was developed which computes the global word-word co-occurrence statistics from a large corpus [62]. There have been other efforts in learning embeddings of documents, paragraphs, or sentences which constituted of performing sentence embeddings rather than word embedding [41, 54, 56].

These models lacked the contextual information of words and were pretty shallow models which resulted in learning a lot of parameters for the model performing a particular downstream task [63]. This led to the development of neural encoders for sentence-level representation. These are called contextual word embeddings as they represent the word meaning depending on the context [64].

[1] Given the time, and compute resource constraints, we perform an evaluation on a small subset of tasks, for a handful of models using a fixed evaluation framework (finetuning).

The first such model was proposed by Dai and Le [19], where they pre-trained an LSTM model with a language modeling (LM) task. They showed that these LSTM models showed many improvements in text classification (TC) tasks, in terms of training convergence as well as task generalization. Seq2Seq models can be significantly improved by unsupervised pre-training as stated by Ramachandran et al. [77]. The weights of both encoders and decoders are initialized with pre-trained weights of two LMs and then fine-tuned on downstream tasks with labeled data. This followed a phenomenal work on pre-training deep LSTM encoders with an attention seq2seq model with machine translation (MT) called CoVe [53]. The output context vectors can improve the performance of a variety of common NLP tasks.

Another technique coined ELMo (Embeddings from Language Models) was pre-training a 2-layer with a bidirectional language model (biLM) task, consisting of a forward LM, and backward LM [64]. These representations showed a huge improvement in a broader range of NLP tasks. These representations are directly supplied to finetune the downstream task model without any training, where the parameters of the main model are still trained from scratch.

ULMFiT (Universal Language Model Fine-tuning) was introduced to fine-tune pre-trained LM for TC tasks and achieve state-of-the-art results on several widely-used TC datasets [35]. This was achieved in 3 steps, pre-training the model, fine-tuning the model on the target data, and fine-tuning the target task using techniques like discriminative fine-tuning, slanted triangular learning rates, and gradual unfreezing.

More recently, we see very deep pre-trained models with the ability to learn universal language representation [22, 71]. This also leads to the introduction of novel pre-training tasks such as self-supervision (see next section) for making pre-trained models capturing more knowledge from large-scale text corpora.

C. *Pre-training Tasks*

Pre-training tasks are essential to learning the universal representation of language. They are categorized into three types: supervised [17], unsupervised [5] and self-supervised learning (SSL) [108]. With these broad categories, we can look at the actual pre-training tasks. For NLP, datasets of most supervised tasks are not large enough to train good pre-trained models. So, we mainly focus on unsupervised tasks and self-supervised. Here, unsupervised learning refers to learning representations from unlabeled data such as densities, latent representations, and clusters. While SSL refers to a learning representation similar to supervised learning, the training labels are generated automatically. E.g. Masked Language Modeling (MLM) is an SSL task where the model tries to predict the masked words in a sentence given the rest of the words (context).

Firstly, the most common unsupervised learning task is probabilistic language modeling (LM). These models learn the representation by getting information from the left context only by training a NN using Maximum likelihood estimation (MLE). An alternative improvement over this is bidirectional-LM i.e. forward, and backward LM [3].

Masked LM (MLM) is another type of pre-training task that masks out some parts of the sentence and tries to predict these mask values inevitably learning the sentence representation to be used for the downstream task [22, 96]. This was an improvement over the previous task i.e. learning forward as well as backward context in a single pass. MLM task is usually helpful in classification-based downstream tasks as it only learns word representation as a neural encoder followed by a classification head [22]. To improve this, we can use encoder-decoder-based architecture for MLM tasks, where the encoder is fed masked sequences, and the decoder sequentially generates the masked tokens, in an auto-regressive manner (Seq2Seq MLM task) [73]. Other improvements over MLM tasks include dynamic masking [51], learning all the previous types of LM tasks all altogether [24], translation LM [37] ...etc. These types of tasks are referred to as enhanced MLM tasks [70].

Despite their huge success, it has been claimed that there is a knowledge gap between pre-training and finetuning because of the absence of $[MASK]$ tokens while learning downstream tasks [103]. To overcome this, Permuted LMs were introduced, where a random permutation of an input sequence is sampled while keeping the original position intact, and such that the model tries to predict some of the target words in this permutation using a two-stream self-attention for position-aware, and target-aware representations.

Next Sentence Prediction (NSP) is another kind of sentence completion task where given two sentences, the model has to predict whether they are continuous segments or not [22]. After BERT's huge success using NSP, there have been a lot of debates on the usefulness of NSP task [37, 42, 51, 103]. Lan et al. introduce Sentence Order Prediction (SOP) as a replacement for NSP [42] reasoning with the difficulty of the task as compared to NSP.

Newer encoder-decoder architectures propose a Denoising Autoencoder (DAE) based pre-training task where a partially corrupted input is taken as input, and the model tries to recover the original undistorted sequence [45, 73, 79]. Corruption of tasks can be done in several different ways such as token deletion, text infilling, sentence permutation, and document rotation [45]. Other advanced techniques include contrastive learning (CTL) [2], which is learning from comparison [31], and Replaced Toke Detection (RTD) [15] where the model has to predict which token is replaced given context[2].

D. *Adaptation to Downstream Tasks*

After learning universal language representation from a large corpus, effective adaptation to the downstream task is another challenging problem.

One of the first adaptation methods is using Transfer learning i.e. to pre-train on a certain task and finetune on

[2] These methods can be discussed in detail in a follow-up larger survey paper as future work.

the target task [60]. This includes techniques such as cross-lingual learning,multi-task learning, domain adaptation, and sequential transfer learning [70]. In general, while performing transfer learning, we need to take care of choosing appropriate pre-training tasks, model architectures, and prospective corpus [72]. As new models emerge with large capacities, another crucial parameter to consider is selecting which layers of the model to use for the downstream task. This can be further divided into either using only embedding layers, the top-most layer, or considering using all the layers concatenated into each other [6, 54, 64].

Further categorizing transfer learning is either feature extraction from frozen pre-trained models or *fine*-tuning pre-trained models [65]. The former requires a large amount of complex task-specific architecture. Thus, fine-tuning has become the common adaptation method for pre-trained models.

There are loads of different types of fine-tuning. One of them is called two-stage tuning, where we leverage a middle stage between pre-training and fine-tuning. The first stage consists of transferring a pre-trained model to an intermediate task using fine-tuning. The second stage consists of the transferred model being further fine-tuned on the target task. This has been used to bring improvements to a large number of existing models like GPT, and BERT [28, 47, 67, 92].

Multi-task fine-tuning is another technique where models are trained on multiple tasks simultaneously, and this method shows hand-in-hand accordance with a pre-training task [50]. Other fine-tuning methods use techniques like model distillation [81], gradual unfreezing [35], planned sequential unfreezing by gradually unfreezing the task-specific layer, followed by the hidden layers of the pre-trained models [13]. All these methods show that fine-tuning can extract task-specific information succinctly. The root problem of fine-tuning is that it is quite frangible: a small amount of hyper-parameter tuning can lead to a huge amount of variations [23].

With the emergence of large pre-trained models trained on enormous amounts of data, alternate methods have been to formulate the downstream task into an MLM task by designing appropriate *prompts* (coined *prompt*-tuning or *in-context*-learning) [25]. This has helped in further helped in decreasing the gap between pre-trained and fine-tuned models. Prompt-based methods have shown tremendous performance in zero-shot [11, 36, 66], few-shot [11, 27, 82]–[84], and fully-supervised settings [46, 49]. Prompt-tuning methods can be categorized into 2 categories i.e. either discrete or continuous prompts.

Discrete prompts are simply some sequence of words inserted into the input text helping the pre-trained model to converge faster for a given downstream task. Earlier works include GPT-3 which proposes in-context learning by concatenating the original input with a task description, and a few examples rather than finetuning the whole pre-trained model achieving on-par performance with the SOTA results on many of the NLP benchmark datasets [11]. Another instance is using discrete prompts with BERT to perform entity prediction (LAMA) without training, showing descent results [83, 84]. Another study shows that manual prompting is sub-optimal in terms of finding prompts as well as the model performance, and thus looks at automatic prompt generation [70]. One such method uses paraphrasing, and mining-based generation, to find optimal patterns that express particular relations. Autoprompt generates optimal prompts using gradient-guided search. [88]. Another method utilizes T5 to automatically generate prompts [27].

Continuous prompting elicits optimal prompts by mapping them in a continuous space rather than discrete one i.e. prompts can be some-other representations rather than words [70]. These continuous prompts are combined with word-type embeddings before passing them through Pre-trained models. Recent studies show that these types of prompts outperform discrete prompts on relation-oriented tasks [69, 106]. Prefix-tuning adds continuous prompt as a prefix to BART, and GPT-2 for downstream tasks [46]. This method as a parameter efficient tuning technique not only achieved similar performance to fully-supervised tuning but also outperformed few-shot fine-tuned models [49]. Another technique named WARP injects trainable prompts in certain parts of the sequence and shows descent performance on common NLP tasks with a fixed pre-trained model [33]. Some newer studies show that prompt-tuning comes close to fine-tuning results as the models scale i.e. when the model size reaches around 100s of billions of parameters, we see that the gap between fine-tuning and prompt-tuning can be closed, as well as the model starts to show emergent behavior [43, 101]. This is also a promising goal toward efficiently serving large-scale pre-trained models while preserving their performance[3].

E. Task Capabilities

In this section, we briefly discuss classical NLP tasks for applications of LLMs. Evaluating LLMs is a continuous problem calling out the development of large-scale benchmarks.

In general, the two most popular benchmarks are GLUE [99], and SuperGLUE [98], where the latter contains harder tasks like co-reference resolution, and question answering. Another paradigm is Question answering (QA), or machine reading comprehension (MRC), which consists of one-shot extractive QA (SQuaD variants) [74, 76], multi-round generative QA [78], and multi-hop QA [104].

Sentiment analysis is another challenging NLP problem, where SST-2 is considered a widely utilized dataset [22]. In the field of information extraction, we have Named Entity Recognition (NER) which plays an important role in other downstream NLP tasks. In this process, the entity information within a sentence is converted into a sequence of labels, where each label corresponds to a single word. The model is then utilized to predict the label for each word [22, 64, 70].

Machine Translation (MT) is another classical NLP task. One of the most common machine translation benchmarks is the WMT (Workshop on Machine Translation) evaluation

[3] By this time, GPT-4 or chatGPT has already taken the world by storm with these techniques as well as human feedback. This unlocks a whole new set of problems to work on in the field of NLP.

campaign, which provides a standard evaluation framework for researchers to compare the performance of different machine translation models across different languages and tasks [1]. Summarization which aims at generating shorter text while still preserving the meaning of the longer text is another important NLP task. A commonly used dataset for this is CNN/DailyMail for extractive summarization tasks [58].

III. Model Catalog

Now that we have looked at the background of different components or LLMs, let's look at a few of the models with a macroscopic view of all their properties such as model pre-training task, training framework, model architecture, dataset details, parameter count, adaptation to downstream tasks, task completion capabilities, and the root research questions that each of these models aim to answer[4].

A. Roberta

RoBERTa was first published in July 2019 by researchers from the University of Washington and Google[5] [51]. Roberta is an extension of BERT that improves its performance by optimizing the training procedure and using more data. RoBERTa belongs to the BERT family of models that use a Transformer encoder architecture and a masked language modeling (MLM) objective with dynamic masking. RoBERTa can be applied to the same natural language understanding tasks as BERT, such as question answering, natural language inference, and sentiment analysis. RoBERTa has two variants: roberta-base and roberta-large, which use the BERT-base and BERT-large architectures respectively. RoBERTa-base has 125 million parameters and can be downloaded from while RoBERTa-large has 355 million parameters; and was pre-trained on the same corpora as BERT (BooksCorpus and English Wikipedia) plus three additional sources: CC News, OpenWebText, and Stories, totaling 33 billion tokens.

B. Deberta

DeBERTa was first published in June 2020 by researchers from Microsoft[6] [34]. DeBERTa enhances the BERT and RoBERTa models with two techniques: a disentangled attention mechanism and an enhanced mask decoder. A disentangled attention mechanism is a technique where each word is represented by two vectors that encode its content and position separately, and the attention weights are computed using disentangled matrices on their contents and relative positions separately. DeBERTa belongs to the BERT family of models that use a Transformer-based encoder architecture and a masked language modeling objective with dynamic masking. DeBERTa can be applied to various NLU tasks, such as question answering, natural language inference, and sentiment analysis. DeBERTa has three variants: DeBERTa-base, DeBERTa-large, and DeBERTa-xlarge, which have 144 million, 350 million, and 700 million parameters respectively. DeBERTa was pre-trained on the same corpora as BERT (BooksCorpus and English Wikipedia) plus an additional source: RealNews, totaling 38 billion tokens.

C. GPT-2

GPT-2 was first published in February 2019 by researchers from OpenAI[7] [72]. GPT-2 belongs to the GPT family of models that use a Transformer decoder architecture and a masked language modeling objective with dynamic masking. The improvements over the previous GPT model are increased context length from 512 to 1024, and layer-normalization moved to the input of each sub-layer. GPT-2 can be applied to various natural language generation tasks, such as text summarization, text translation, and text completion, as well as can be finetuned for other NLP tasks. GPT-2 has four variants: GPT-2-small, GPT-2-medium, GPT-2-large, and GPT-2-XL, which have 117 million, 345 million, 762 million, and 1.5 billion parameters respectively. GPT-2 was pre-trained on a large corpus of web pages called WebText, which contains 40 GB of text and 8 million documents scraped from websites with high Reddit karma scores.

D. Transformer-XL

Transformer XL was first published in January 2019 by researchers from Carnegie Mellon University and Google[8] [20]. Transformer XL extends the Transformer model with a segment-level recurrence mechanism and a relative positional encoding scheme. Transformer XL belongs to the family of models that use a Transformer decoder architecture and a masked language modeling objective. Transformer XL can be applied to general language tasks, such as text generation, text summarization, and text understanding. Transformer XL has 355 million parameters and was pre-trained on different training datasets depending on the experiments, but the baseline dataset was Wikitext-103, which contains 103 million tokens of Wikipedia articles. Transformer XL introduces relatively positioned embeddings that enable longer-context attention when compared to the vanilla Transformer model, which can only attend to a fixed-length context. This allows Transformer XL to capture long-range dependencies and improve the quality and coherence of the generated text.

E. Bart

BART was first published in October 2019 by researchers from Facebook[9] [45]. BART uses a Transformer-based encoder-decoder architecture and a denoising autoencoder (DAE) objective. BART belongs to the family of models that combine ideas from BERT and GPT, where the encoder is similar to BERT and the decoder is similar to GPT. BART can be seen as a generalization of BERT and GPT in that it uses a bidirectional encoder and a left-to-right decoder. BART uses a span corruption method to create noisy inputs for the

[4] Because of the page limit and limited amount of time, we have only included the model catalog for the models that we tested. More models can be added down the road on a larger survey paper.

[5] https://github.com/facebookresearch/fairseq/tree/main/examples/roberta

[6] https://github.com/microsoft/DeBERTa

[7] https://github.com/openai/gpt-2

[8] https://github.com/kimiyoung/transformer-xl

[9] https://github.com/facebookresearch/fairseq/blob/main/examples/bart

DAE objective, which randomly replaces spans of text with either a special mask token or the original text. This allows BART to learn from both corrupted and uncorrupted tokens and improve the quality and diversity of the generated text. BART can be applied to mostly text generation tasks but also some text understanding tasks, such as text summarization, text translation, and text classification. BART has 10% more parameters than BERT and was pre-trained on the same corpora as RoBERTa, which contains 160 GB of text from news, books, and stories.

F. T5

T5 was first published in October 2019 by researchers from Google[10] [73]. T5 uses a unified text-to-text framework for both pre-training and fine-tuning. T5 belongs to the family of models that use a Transformer encoder-decoder architecture and a denoising autoencoder (DAE) objective. T5 extends the original Transformer model with some additions such as relative positional embeddings like Transformer XL, which allow the model to attend to longer contexts. T5 also uses a text-to-text format for all inputs and outputs, which simplifies the data preprocessing and task adaptation steps for fine-tuning as well. T5 uses a span corruption method to create noisy inputs for the DAE objective, which randomly replaces spans of text with a special mask token. T5 can be applied to general language tasks, including machine translation, question answering, abstractive summarization, and text classification. T5 has five variants: T5-small, T5-base, T5-large, T5-3B, and T5-11B, which have 60 million, 220 million, 770 million, 3 billion, and 11 billion parameters respectively. It was pre-trained on the Colossal Clean Crawled Corpus (C4), which is a cleaned-up version of the Common Crawl dataset containing 750 GB of text.

IV. Experiment and Analysis

A. Dataset Details

To perform a complete evaluation, we choose multiple common NLP tasks covered by the SuperGLUE dataset [98]. Each model is pre-trained on their respective pre-training datasets. We choose the Super Glue dataset [98] for finetuning with the respective tasks:

- Coreference Resolution using Winograd Schema Challenge (WSC) [44] - The system needs to identify the appropriate antecedent for a pronoun in a sentence that also includes a series of noun phrases. The schemas are constructed to require commonsense reasoning from the model to correctly solve the problem.
- Boolean Question Answering (BoolQ) [14]- This is an evaluation task for question answering, where each example includes a brief passage and a yes/no question relating to that passage. The questions are submitted anonymously and voluntarily by users of Google search, and they are paired with a paragraph from a relevant Wikipedia article that contains the answer. Accuracy is the metric used to assess performance on this dataset.
- Natural Language Inference (RTE) [9, 18, 29, 32]- The task involves determining whether a given pair of premises and hypotheses entail each other or not. This is a combination of five different datasets (RTE 1-5). The evaluation of this task is based on the accuracy of the predictions.
- Word sense disambiguation (WIC) [68]- This is a binary classification task for word sense disambiguation, where the goal is to determine whether a polysemous word (a word with multiple meanings) appearing in two given text snippets is used in the same sense in both sentences. The sentences are constructed using WordNet [57], VerbNet [85], and Wiktionary. The evaluation of the task is measured using the accuracy metric.
- Three-class textual entailment (CB) [21]- The dataset contains short texts with at least one sentence containing an embedded clause. Each embedded sentence is annotated with the degree to which the author appears to believe the clause is true, resulting in a three-class textual entailment task for examples from Wall Street Journal, British National Corpus fiction, and Switchboard. Each example consists of a premise containing the embedded clause and a hypothesis, which is the extraction of that clause. The dataset is a subset of the larger dataset with an inter-annotation score of over 80%. Due to the dataset's imbalance with fewer neutral examples, it is evaluated using both accuracy and F1 score.
- Choice of Plausible Alternatives (COPA) [80]- This is a task focused on causal reasoning, where a given sentence or premise is presented, and the model must identify either the cause or effect from two possible choices. The examples for this task are carefully crafted and selected from blogs and photography-related encyclopedias. The evaluation of this task is based on accuracy.

More details about the number of training examples, validation examples, task type, metrics, and text sources for each dataset are described in Table I. Each task has a training set and a dev set. We use the training set to finetune all the models, and the dev set to measure the performance of each model.

B. Model details

The models that we finetuned are described in Table II. We used 2 models from each architecture family, i.e. encoder-only models (roberta-large, deberta-large), decoder-only models (gpt2-medium, transformer-xl), and encoder-decoder models (bart-large, t5-large). The table also shows how most of the model size is comparable except the t5 model. As for the size of the model, we used the largest possible pre-trained models available for each model family to maximize the pre-training gains.

C. Experiment Setup

For training, we used a sequence classification head over each of these models except the t5 model. Basically, we framed

[10] https://github.com/google-research/text-to-text-transfer-transformer

TABLE I: The tasks included in each dataset. WSD stands for word sense disambiguation, NLI is natural language inference, coref. is coreference resolution, and QA is question answering.

Corpus	Train Set	Dev Set	Task	Metrics	Text Sources
BoolQ	9,427	3,270	QA	acc.	Google queries, Wikipedia
RTE	2,500	278	NLI	acc.	news, Wikipedia
COPA	400	100	QA	acc.	blogs, photography encyclopedia
WIC	6,000	638	WSD	acc.	WordNet, VerbNet, Wiktionary
WSC	554	104	coref.	acc.	fiction books
CB	250	57	NLI	acc./F1	various

TABLE II: The details of each model used for training such as the model size (parameters), learning rate, maximum epochs it was fine-tuned for, weight decay, batch size, and sequence length. Here M in the parameter column is million.

Model	Parameters	Learning Rate	Max Epochs	Batch Size	Weight Decay	Sequence Length
roberta-large	355M	3e-5	10	32	0.1	512
deberta-large	350M	1e-5	10	16	0.01	512
gpt2-medium	355M	3e-5	10	8	0.01	1024
transformer-xl	355M	2.5e-4	10	16	0.01	512
bart-large	400M	1e-5	10	16	0.01	512
t5-large	770M	1e-3	10	8	0.0	512

each task as a classification task and performed training on each of the models accordingly making a compatible study of the model. The Table II also describes all the hyperparameters that we tried out for tuning the model which includes learning rate, maximum epochs for which we trained the model, batch size used for training the model, weight decay to add regularization for large models on small datasets, and gradient clipping [61] to avoid exploding gradients (we tried out a lot of other hyperparameters, and these were the ones that worked the best from our experiments).

All the parameters for respective models remain the same for finetuning on each dataset (thus we have 6 models for each model-family 1 for each task). On top of this, we used Adam with weight decay (AdamW) [52] as our optimizer. As it was a classification task, we used the cross-entropy loss for each task. We used a linear learning rate scheduler as well. To avoid overfitting, we monitor the training loss as well as validation loss after every epoch and save the model if it shows the best loss/accuracy so far. For t5, the model that we loaded from transformers is already finetuned on the SuperGLUE dataset on a single pass as a conditional generator model using a multi-task objective, so we only performed an evaluation on that model based on the paper's discrete prompt design. As t5 was trained on the SuperGLUE task as a whole, we only have one single model for t5 that we use for all the select SuperGLUE tasks.

For evaluation, we used the dev set whose results can be found in Table III. For each model except the t5 model, we performed the sequence classification evaluation as and how it was finetuned on the SuperGLUE dataset for each model and each task. T5 was trained on conditional generation for all the SuperGLUE tasks, we also use a similar format for evaluation, where the input is formatted with discrete prompts, and the outputs are cross-verified according to how the original t5 paper's description. This formatting is discussed with an example for each task in Table IV.

All our experiments were carried out on an A100 80GB GPU, and 15GB CPU on HiperGator3.0[11]. The source code was written in Python 3.8+[12], specifically, the model code itself was written using the transformers[13], and the PyTorch package[14]. We used the Accelerate library to handle GPU training. All the dependencies for different libraries for training and evaluation were handled using the mamba (similar to Anaconda)[15]. We tracked different versions of the codebase using Git[16], and GitHub[17] and used Visual Studio[18] as a source-code editing platform. For submitting the jobs to HiperGator, we used the Slurm workload manager[19].

D. Analysis

Looking at the results from Table III, we can clearly see that the Deberta models show the best performance on BoolQ, WIC, and WSC, while the T5 model shows the best performance on RTE, COPA, and CB. One of the patterns here is that both RTE, and CB are entailment tasks, and thus it means that T5 is better suited for natural language inferencing tasks in this particular scenario. Another thing to note here is that Deberta is able to learn robust representation for low-resource tasks like WIC, and WSC.

Another thing to notice here is that decoder-only models didn't perform very well because they might need better encoder representations, as well as the use of a causal generation-based task might be better for these models. Also, encoder-decoder models overall perform better than encoder-only models, and decoder-only models in the sense that they can learn representations using the encoder, and generate outputs using causal generators, this is especially true for how the T5 was

[11] https://www.rc.ufl.edu/about/hipergator/
[12] https://www.python.org/
[13] https://huggingface.co/
[14] https://pytorch.org/
[15] https://mamba.readthedocs.io/en/latest/
[16] https://git-scm.com/
[17] https://github.com/
[18] https://code.visualstudio.com/
[19] https://slurm.schedmd.com/documentation.html

TABLE III: Performance of all the models on different tasks where the highlighted models performed the best on each task.

Dataset / Model	Roberta	Deberta	GPT-2	Transformer-XL	Bart	T5
BoolQ (acc.)	0.6217	**0.8685**	0.7695	0.6714	0.85107	0.7877
RTE (acc.)	0.527	0.8591	0.7184	0.570	0.8519	**0.8592**
COPA (acc.)	0.55	0.58	0.6	0.57	0.56	**0.66**
WIC (acc.)	0.5	**0.7116**	0.6912	0.5360	0.6834	0.6914
WSC (acc.)	0.6302	**0.6347**	0.6442	0.6340	0.6346	0.5194
CB (acc./F1)	0.8081 / 0.765	0.6785 / 0.4740	0.7857 / 0.6398	0.7321 / 0.5113	0.6785 / 0.4700	**0.875 / 0.7854**

TABLE IV: Discrete prompt format for evaluation of T5 model for each dataset. Here we replace the placeholder <text> with the corresponding text from the dataset. For COPA, the question format is What was the cause of this? if it is cause, and What happened as a result of this? if it is an effect type relationship respectively. For WSC we highlight the second-word span whose relationship needs to be found out with *. For output formats, we show what is the actual output of the model, followed by its actual label i.e. <output> → <label>

Dataset	Input Format	Output Format
BoolQ	boolq context: <text> question: <text>	false → 0, true → 1
RTE	rte sentence1: <text> sentence2: <text>	entailment → 0, not_entailment → 1
COPA	copa choice1: <text> choice2: <text> premise: <text> question: <text>	choice1 → 0, choice2 → 1
WIC	wic sentence1: <text> sentence2: <text> word: <text>	false → 0, true → 1
WSC	wsc sentence: <text *span2_word* text>	acceptable → 0, not_acceptable → 1
CB	cb hypothesis: <text> premise: <text>	entailment → 0, contradiction → 1, neutral → 2

finetuned. This is the reason why T5 performs better than Bart in most of the tasks.

The most difficult of all the tasks were COPA, and WSC because of the low amount of labeled data, as well as the task difficulty. We can see that for the plausible alternatives tasks, decoder-based architectures learn better representations, and for coreference resolution, most of the models seems to be learning similar representation showing the inherent intricacy of the task.

Note that for each model, the original paper shows better performance in most of the tasks because they weren't trained directly on the same head as we did, for each task, each of the models was hand-crafted with a lot of custom heads, as well as using different kinds of transfer learning framework to get good performance.

All of this shows how each model is capable of a particular kind of task, and which type of models can be used for a particular downstream task accordingly. Of course, further analysis, and ablations can reveal the important task capabilities of each model.

E. Future Work

In future work, we can experiment with a larger roster of models and more robust datasets like Big-bench [91] to truly form a large survey with a practical task capability study. Another thing to explore is to try different training frame-works mentioned in a few of the papers such as an in-depth survey of in-context learning (this can be further broken down into different types of prompting techniques like Chain-of-thought prompting (CoT) [102], Auto-CoT [105], Self-Consistency prompting [100], ...etc), using different pre-training criteria like UL2 [95], using Megatron style [89] training for large models, instruction finetuning on smaller models to see the effects of human preference [59], study the root causes of emergent abilities on large-scale LLMs, interpret-ability and reliability of pre-trained models ...etc.

Another possible direction to tackle is the regulation of these models because as the increases in size, and the invention of finer techniques, even a 1B parameter model could have massive societal repercussions. Lastly, with the advent of new models being trained on all the datasets of the world, a thorough contamination analysis, as well as the development of new datasets to re-challenge these models can be done [91]. This also calls out the purpose of the application of LLMs in real life and how we can compress them such that they can serve a large user base as well as retain their original prediction capabilities with minimum latency.

V. Conclusion

In this paper, we have conducted a systematic survey on language models that aims to establish a connection among various models based on multiple criteria, such as representation learning, model size, task capabilities, research questions, and practical task capabilities. We have examined different types of language models, ranging from shallow word embeddings to deep contextual encoders, and analyzed their pre-training tasks, training frameworks, adaptation methods, and evaluation benchmarks. We have also performed experiments on a subset of SuperGLUE tasks using six representative models from different architecture families and compared their performance and efficiency. Our results reveal that different models have different advantages and disadvantages depending on the task and the data. Furthermore, we propose some future directions that can be explored to enhance the robustness and comprehensiveness of this survey. We hope that our survey can provide a historical perspective on the development of language modeling and inspire future research directions in this field.

References

[1] machinetranslate.org. https://machinetranslate.org/. Accessed: 2022-05-12.

[2] Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*, 2019.

[3] Alexei Baevski, Sergey Edunov, Yinhan Liu, Luke Zettlemoyer, and Michael Auli. Cloze-driven pretraining of self-attention networks. *arXiv preprint arXiv:1903.07785*, 2019.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[5] Horace B Barlow. Unsupervised learning. *Neural computation*, 1(3):295–311, 1989.

[6] Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. What do neural machine translation models learn about morphology? *arXiv preprint arXiv:1704.03471*, 2017.

[7] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, August 2013. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

[8] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. *Advances in neural information processing systems*, 13, 2000.

[9] Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. The fifth pascal recognizing textual entailment challenge. In *TAC*. Citeseer, 2009.

[10] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146, 2017.

[11] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D 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 Ziegler, Jeffrey Wu, Clemens Winter, Chris 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. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[12] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[13] Alexandra Chronopoulou, Christos Baziotis, and Alexandros Potamianos. An embarrassingly simple approach for transfer learning from pretrained language models. *arXiv preprint arXiv:1902.10547*, 2019.

[14] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions, May 2019. arXiv:1905.10044 [cs].

[15] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators, March 2020. arXiv:2003.10555 [cs].

[16] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(ARTICLE):2493–2537, 2011.

[17] Pádraig Cunningham, Matthieu Cord, and Sarah Jane Delany. Supervised learning. *Machine learning techniques for multimedia: case studies on organization and retrieval*, pages 21–49, 2008.

[18] Ido Dagan, Oren Glickman, and Bernardo Magnini. The PASCAL Recognising Textual Entailment Challenge. In Joaquin Quiñonero-Candela, Ido Dagan, Bernardo Magnini, and Florence d'Alché Buc, editors, *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, Lecture Notes in Computer Science, pages 177–190, Berlin, Heidelberg, 2006. Springer.

[19] Andrew M Dai and Quoc V Le. Semi-supervised sequence learning. *Advances in neural information processing systems*, 28, 2015.

[20] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context, June 2019. arXiv:1901.02860 [cs, stat].

[21] Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23, pages 107–124, 2019.

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, May 2019. arXiv:1810.04805 [cs].

[23] Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*, 2020.

[24] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation. *Advances in neural information processing systems*, 32, 2019.

[25] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. A Survey for In-context Learning, December 2022. arXiv:2301.00234 [cs].

[26] Dumitru Erhan, Aaron Courville, Yoshua Bengio, and Pascal Vincent. Why does unsupervised pre-training help deep learning? In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 201–208. JMLR Workshop and Conference Proceedings, 2010.

[27] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*, 2020.

[28] Siddhant Garg, Thuy Vu, and Alessandro Moschitti. Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7780–7788, 2020.

[29] Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. The Third PASCAL Recognizing Textual Entailment Challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague, June 2007. Association for Computational Linguistics.

[30] Alex Graves and Alex Graves. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, pages 37–45, 2012.

[31] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010.

[32] R Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. The second pascal recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, volume 7, 2006.

[33] Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. Warp: Word-level adversarial reprogramming. *arXiv preprint arXiv:2101.00121*, 2021.

[34] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. DeBERTa: Decoding-enhanced BERT with Disentangled Attention, October 2021. arXiv:2006.03654 [cs].

[35] Jeremy Howard and Sebastian Ruder. Universal Language Model Fine-tuning for Text Classification, May 2018. arXiv:1801.06146 [cs, stat].

[36] Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438, 2020.

[37] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, 2020.

[38] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

[39] Yoon Kim, Yacine Jernite, David Sontag, and Alexander Rush. Character-aware neural language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[40] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[41] Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. *Advances in neural information processing systems*, 28, 2015.

[42] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations, February 2020. arXiv:1909.11942 [cs].

[43] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[44] Hector Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*, 2012.

[45] 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, October 2019. arXiv:1910.13461 [cs, stat].

[46] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[47] Zhongyang Li, Xiao Ding, and Ting Liu. Story ending prediction by transferable bert. *arXiv preprint arXiv:1905.07504*, 2019.

[48] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. Holistic Evaluation of Language Models, November 2022. arXiv:2211.09110 [cs].

[49] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *arXiv preprint arXiv:2103.10385*, 2021.

[50] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-task deep neural networks for natural language understanding. *arXiv preprint arXiv:1901.11504*, 2019.

[51] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach, July 2019. arXiv:1907.11692 [cs].

[52] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[53] Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. Learned in Translation: Contextualized Word Vectors, June 2018. arXiv:1708.00107 [cs].

[54] Oren Melamud, Jacob Goldberger, and Ido Dagan. context2vec: Learning generic context embedding with bidirectional lstm. In *Proceedings of the 20th SIGNLL conference on computational natural language learning*, pages 51–61, 2016.

[55] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[56] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[57] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[58] Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*, 2016.

[59] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[60] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[61] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. Pmlr, 2013.

[62] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[63] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[64] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations, March 2018. arXiv:1802.05365 [cs].

[65] Matthew E Peters, Sebastian Ruder, and Noah A Smith. To tune or not to tune? adapting pretrained representations to diverse tasks. *arXiv preprint arXiv:1903.05987*, 2019.

[66] Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

[67] Jason Phang, Thibault Févry, and Samuel R Bowman. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*, 2018.

[68] Mohammad Taher Pilehvar and Jose Camacho-Collados. WiC: the Word-in-Context Dataset for Evaluating Context-Sensitive Meaning Representations, April 2019. arXiv:1808.09121 [cs].

[69] Guanghui Qin and Jason Eisner. Learning how to ask: Querying lms with mixtures of soft prompts. *arXiv preprint arXiv:2104.06599*, 2021.

[70] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. Pre-trained Models for Natural Language Processing: A Survey. *Science China Technological Sciences*, 63(10):1872–1897, October 2020. arXiv:2003.08271 [cs].

[71] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[72] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[73] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[74] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.

[75] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ Questions for Machine Comprehension of Text, October 2016. arXiv:1606.05250 [cs].

[76] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[77] Prajit Ramachandran, Peter J Liu, and Quoc V Le. Unsupervised pretraining for sequence to sequence learning. *arXiv preprint arXiv:1611.02683*, 2016.

[78] Siva Reddy, Danqi Chen, and Christopher D Manning. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266, 2019.

[79] Adam Roberts, Hyung Won Chung, Anselm Levskaya, Gaurav Mishra, James Bradbury, Daniel Andor, Sharan Narang, Brian Lester, Colin Gaffney, Afroz Mohiuddin, Curtis Hawthorne, Aitor Lewkowycz, Alex Salcianu, Marc van Zee, Jacob Austin, Sebastian Goodman, Livio Baldini Soares, Haitang Hu, Sasha Tsvyashchenko, Aakanksha Chowdhery, Jasmijn Bastings, Jannis Bulian, Xavier Garcia, Jianmo Ni, Andrew Chen, Kathleen Kenealy, Jonathan H. Clark, Stephan Lee, Dan Garrette, James Lee-Thorp, Colin Raffel, Noam Shazeer, Marvin Ritter, Maarten Bosma, Alexandre Passos, Jeremy Maitin-Shepard, Noah Fiedel, Mark Omernick, Brennan Saeta, Ryan Sepassi, Alexander Spiridonov, Joshua Newlan, and Andrea Gesmundo. Scaling Up Models and Data with \$\texttt{t5x}\$ and \$\texttt{seqio}\$, March 2022. arXiv:2203.17189 [cs].

[80] Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. Choice of plausible alternatives: An evaluation of commonsense causal

reasoning. In *AAAI spring symposium: logical formalizations of commonsense reasoning*, pages 90–95, 2011.

[81] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, February 2020. arXiv:1910.01108 [cs].

[82] Teven Le Scao and Alexander M Rush. How many data points is a prompt worth? *arXiv preprint arXiv:2103.08493*, 2021.

[83] Timo Schick and Hinrich Schütze. Exploiting cloze questions for few shot text classification and natural language inference. *arXiv preprint arXiv:2001.07676*, 2020.

[84] Timo Schick and Hinrich Schütze. It's not just size that matters: Small language models are also few-shot learners. *arXiv preprint arXiv:2009.07118*, 2020.

[85] Karin Kipper Schuler. *VerbNet: A broad-coverage, comprehensive verb lexicon*. University of Pennsylvania, 2005.

[86] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

[87] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

[88] Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020.

[89] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism, March 2020. arXiv:1909.08053 [cs].

[90] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[91] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ambrose Slone, Ameet Rahane, Anantharaman S. Iyer, Anders Andreassen, Andrea Madotto, Andrea Santilli, Andreas Stuhlmüller, Andrew Dai, Andrew La, Andrew Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakaş, B. Ryan Roberts, Bao Sheng Loe, Barret Zoph, Bartłomiej Bojanowski, Batuhan Özyurt, Behnam Hedayatnia, Behnam Neyshabur, Benjamin Inden, Benno Stein, Berk Ekmekci, Bill Yuchen Lin, Blake Howald, Cameron Diao, Cameron Dour, Catherine Stinson, Cedrick Argueta, César Ferri Ramírez, Chandan Singh, Charles Rathkopf, Chenlin Meng, Chitta Baral, Chiyu Wu, Chris Callison-Burch, Chris Waites, Christian Voigt, Christopher D. Manning, Christopher Potts, Cindy Ramirez, Clara E. Rivera, Clemencia Siro, Colin Raffel, Courtney Ashcraft, Cristina Garbacea, Damien Sileo, Dan Garrette, Dan Hendrycks, Dan Kilman, Dan Roth, Daniel Freeman, Daniel Khashabi, Daniel Levy, Daniel Moseguí González, Danielle Perszyk, Danny Hernandez, Danqi Chen, Daphne Ippolito, Dar Gilboa, David Dohan, David Drakard, David Jurgens, Debajyoti Datta, Deep Ganguli, Denis Emelin, Denis Kleyko, Deniz Yuret, Derek Chen, Derek Tam, Dieuwke Hupkes, Diganta Misra, Dilyar Buzan, Dimitri Coelho Mollo, Diyi Yang, Dong-Ho Lee, Ekaterina Shutova, Ekin Dogus Cubuk, Elad Segal, Eleanor Hagerman, Elizabeth Barnes, Elizabeth Donoway, Ellie Pavlick, Emanuele Rodola, Emma Lam, Eric Chu, Eric Tang, Erkut Erdem, Ernie Chang, Ethan A. Chi, Ethan Dyer, Ethan Jerzak, Ethan Kim, Eunice Engefu Manyasi, Evgenii Zheltonozhskii, Fanyue Xia, Fatemeh Siar, Fernando Martínez-Plumed, Francesca Happé, Francois Chollet, Frieda Rong, Gaurav Mishra, Genta Indra Winata, Gerard de Melo, Germán Kruszewski, Giambattista Parascandolo, Giorgio Mariani, Gloria Wang, Gonzalo Jaimovitch-López, Gregor Betz, Guy Gur-Ari, Hana Galijasevic, Hannah Kim, Hannah Rashkin, Hannaneh Hajishirzi, Harsh Mehta, Hayden Bogar, Henry Shevlin, Hinrich Schütze, Hiromu Yakura, Hongming Zhang, Hugh Mee Wong, Ian Ng, Isaac Noble, Jaap Jumelet, Jack Geissinger, Jackson Kernion, Jacob Hilton, Jaehoon Lee, Jaime Fernández Fisac, James B. Simon, James Koppel, James Zheng, James Zou, Jan Kocoń, Jana Thompson, Jared Kaplan, Jarema Radom, Jascha Sohl-Dickstein, Jason Phang, Jason Wei, Jason Yosinski, Jekaterina Novikova, Jelle Bosscher, Jennifer Marsh, Jeremy Kim, Jeroen Taal, Jesse Engel, Jesujoba Alabi, Jiacheng Xu, Jiaming Song, Jillian Tang, Joan Waweru, John Burden, John Miller, John U. Balis, Jonathan Berant, Jörg Frohberg, Jos Rozen, Jose Hernandez-Orallo, Joseph Boudeman, Joseph Jones, Joshua B. Tenenbaum, Joshua S. Rule, Joyce Chua, Kamil Kanclerz, Karen Livescu, Karl Krauth, Karthik Gopalakrishnan, Katerina Ignatyeva, Katja Markert, Kaustubh D. Dhole, Kevin Gimpel, Kevin Omondi, Kory Mathewson, Kristen Chiafullo, Ksenia Shkaruta, Kumar Shridhar, Kyle McDonell, Kyle Richardson, Laria Reynolds, Leo Gao, Li Zhang, Liam Dugan, Lianhui Qin, Lidia Contreras-Ochando, Louis-Philippe Morency, Luca Moschella, Lucas Lam, Lucy Noble, Ludwig Schmidt, Luheng He, Luis Oliveros Colón, Luke Metz, Lütfi Kerem Şenel, Maarten Bosma, Maarten Sap, Maartje ter Hoeve, Maheen Farooqi, Manaal Faruqui, Mantas Mazeika, Marco Baturan, Marco Marelli, Marco Maru, Maria Jose Ramírez Quintana, Marie Tolkiehn, Mario Giulianelli, Martha Lewis, Martin Potthast, Matthew L. Leavitt, Matthias Hagen, Mátyás Schubert, Medina Orduna Baitemirova, Melody Arnaud, Melvin McElrath, Michael A. Yee, Michael Cohen, Michael Gu, Michael Ivanitskiy, Michael Starritt, Michael Strube, Michał Swędrowski, Michele Bevilacqua, Michihiro Yasunaga, Mihir Kale, Mike Cain, Mimee Xu, Mirac Suzgun, Mo Tiwari, Mohit Bansal, Moin Aminnaseri, Mor Geva, Mozhdeh Gheini, Mukund Varma T, Nanyun Peng, Nathan Chi, Nayeon Lee, Neta Gur-Ari Krakover, Nicholas Cameron, Nicholas Roberts, Nick Doiron, Nikita Nangia, Niklas Deckers, Niklas Muennighoff, Nitish Shirish Keskar, Niveditha S. Iyer, Noah Constant, Noah Fiedel, Nuan Wen, Oliver Zhang, Omar Agha, Omar Elbaghdadi, Omer Levy, Owain Evans, Pablo Antonio Moreno Casares, Parth Doshi, Pascale Fung, Paul Pu Liang, Paul Vicol, Pegah Alipoormolabashi, Peiyuan Liao, Percy Liang, Peter Chang, Peter Eckersley, Phu Mon Htut, Pinyu Hwang, Piotr Miłkowski, Piyush Patil, Pouya Pezeshkpour, Priti Oli, Qiaozhu Mei, Qing Lyu, Qinlang Chen, Rabin Banjade, Rachel Etta Rudolph, Raefer Gabriel, Rahel Habacker, Ramón Risco Delgado, Raphaël Millière, Rhythm Garg, Richard Barnes, Rif A. Saurous, Riku Arakawa, Robbe Raymaekers, Robert Frank, Rohan Sikand, Roman Novak, Roman Sitelew, Ronan LeBras, Rosanne Liu, Rowan Jacobs, Rui Zhang, Ruslan Salakhutdinov, Ryan Chi, Ryan Lee, Ryan Stovall, Ryan Teehan, Rylan Yang, Sahib Singh, Saif M. Mohammad, Sajant Anand, Sam Dillavou, Sam Shleifer, Sam Wiseman, Samuel Gruetter, Samuel R. Bowman, Samuel S. Schoenholz, Sanghyun Han, Sanjeev Kwatra, Sarah A. Rous, Sarik Ghazarian, Sayan Ghosh, Sean Casey, Sebastian Bischoff, Sebastian Gehrmann, Sebastian Schuster, Sepideh Sadeghi, Shadi Hamdan, Sharon Zhou, Shashank Srivastava, Sherry Shi, Shikhar Singh, Shima Asaadi, Shixiang Shane Gu, Shubh Pachchigar, Shubham Toshniwal, Shyam Upadhyay, Shyamolima, Debnath, Siamak Shakeri, Simon Thormeyer, Simone Melzi, Siva Reddy, Sneha Priscilla Makini, Soo-Hwan Lee, Spencer Torene, Sriharsha Hatwar, Stanislas Dehaene, Stefan Divic, Stefano Ermon, Stella Biderman, Stephanie Lin, Stephen Prasad, Steven T. Piantadosi, Stuart M. Shieber, Summer Misherghi, Svetlana Kiritchenko, Swaroop Mishra, Tal Linzen, Tal Schuster, Tao Li, Tao Yu, Tariq Ali, Tatsu Hashimoto, Te-Lin Wu, Théo Desbordes, Theodore Rothschild, Thomas Phan, Tianle Wang, Tiberius Nkinyili, Timo Schick, Timofei Kornev, Timothy Telleen-Lawton, Titus Tunduny, Tobias Gerstenberg, Trenton Chang, Trishala Neeraj, Tushar Khot, Tyler Shultz, Uri Shaham, Vedant Misra, Vera Demberg, Victoria Nyamai, Vikas Raunak, Vinay Ramasesh, Vinay Uday Prabhu, Vishakh Padmakumar, Vivek Srikumar, William Fedus, William Saunders, William Zhang, Wout Vossen, Xiang Ren, Xiaoyu Tong, Xinran Zhao, Xinyi Wu, Xudong Shen, Yadollah Yaghoobzadeh, Yair Lakretz, Yangqiu Song, Yasaman Bahri, Yejin Choi, Yichi Yang, Yiding Hao, Yifu Chen, Yonatan Belinkov, Yu Hou, Yufang Hou, Yuntao Bai, Zachary Seid, Zhuoye Zhao, Zijian Wang, Zijie J. Wang, Zirui Wang, and Ziyi Wu. Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models, June 2022. arXiv:2206.04615 [cs, stat].

[92] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune bert for text classification? In *Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18*, pages 194–206. Springer, 2019.

[93] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence

learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

[94] Kai Sheng Tai, Richard Socher, and Christopher D Manning. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*, 2015.

[95] Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Denny Zhou, Neil Houlsby, and Donald Metzler. UL2: Unifying Language Learning Paradigms, February 2023. arXiv:2205.05131 [cs].

[96] Wilson L Taylor. "cloze procedure": A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433, 1953.

[97] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[98] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[99] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

[100] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

[101] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent Abilities of Large Language Models, October 2022. arXiv:2206.07682 [cs].

[102] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models, January 2023. arXiv:2201.11903 [cs].

[103] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. XLNet: Generalized Autoregressive Pretraining for Language Understanding, January 2020. arXiv:1906.08237 [cs].

[104] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*, 2018.

[105] Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*, 2022.

[106] Zexuan Zhong, Dan Friedman, and Danqi Chen. Factual probing is [mask]: Learning vs. learning to recall. *arXiv preprint arXiv:2104.05240*, 2021.

[107] Xiaodan Zhu, Parinaz Sobihani, and Hongyu Guo. Long short-term memory over recursive structures. In *International conference on machine learning*, pages 1604–1612. PMLR, 2015.

[108] Xiaojin Jerry Zhu. Semi-supervised learning literature survey. 2005.