Pattern Recognition Letters 136 (2020) 120–126

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

# Multi-task learning for natural language processing in the 2020s: Where are we going?

Joseph Worsham*, Jugal Kalita

*University of Colorado Colorado Springs, 1420 Austin Bluffs Parkway, Colorado Springs, CO 80918, USA*

ARTICLE INFO

*Article history:*
Received 18 November 2019
Revised 1 May 2020
Accepted 26 May 2020
Available online 28 May 2020

*MSC:*
68T05
68T10
68T50
68-02

*Keywords:*
Multi-task learning
Task relationship
Natural language processing

ABSTRACT

Multi-task learning (MTL) significantly pre-dates the deep learning era, and it has seen a resurgence in the past few years as researchers have been applying MTL to deep learning solutions for natural language tasks. While steady MTL research has always been present, there is a growing interest driven by the impressive successes published in the related fields of transfer learning and pre-training, such as BERT, and the release of new challenge problems, such as GLUE and the NLP Decathlon (decaNLP). These efforts place more focus on how weights are shared across networks, evaluate the re-usability of network components and identify use cases where MTL can significantly outperform single-task solutions. This paper strives to provide a comprehensive survey of the numerous recent MTL contributions to the field of natural language processing and provide a forum to focus efforts on the hardest unsolved problems in the next decade. While novel models that improve performance on NLP benchmarks are continually produced, lasting MTL challenges remain unsolved which could hold the key to better language understanding, knowledge discovery and natural language interfaces.



## 1. Introduction

Multi-task learning (MTL) is a collection of techniques intended to improve generalization, strengthen latent representations and enable domain adaptation within the field of machine learning [9]. It has been applied to feed-forward neural networks [9], decision trees [9], random forests [44], Gaussian Processes [8], support-vector machines [16] and, most recently, deep neural networks [34] across a broad range of domains. This includes specific deep learning architectures such as MTL seq2seq models [41] and MTL transformers [20]. It has been shown that under certain circumstances, and with well-crafted tasks, MTL can help models achieve state-of-the-art performance on a range of different tasks [37]. It has also been shown, however, that MTL can be extremely fragile and sensitive to both the selected tasks and the training process which leads to models that significantly under-perform when compared to the best single-task models [2]. While MTL has been a subject of research for multiple decades [34], there still exist a number of unsolved problems, unexplored questions and shortcomings in production systems which are addressed within. This survey will present a condensed summary of the large library of current MTL research applied to natural language processing (NLP) and present a set of goals intended to help highlight the MTL problems that we should strive to solve in the next decade.

## 2. Characterizing multi-task learning

MTL introduces additional training objectives to a learning system to bias the learner with a broader understanding through solving related tasks. The end-goal is to improve performance on a set of primary tasks through the inductive bias introduced by the additional tasks [9]. The set of primary tasks are referred to as the target task set, and additional tasks, which are used to improve performance on the target set, belong to the auxiliary task set. While this is the standard approach [34], others have also designed MTL models with no auxiliary focusing on competitively solving all the tasks jointly [23].

In practice, MTL is closely related to Transfer Learning (TL) [5], as the goal of each is to improve the performance of a target task or domain through the use of related tasks and domains. A task is defined as a specific operational capability, such as Part of Speech Tagging or Question Answering. Tasks traditionally do not share the same output features or solution space. A domain is a certain feature space and underlying data generating process. When working with TL and MTL, commonly, different domains share the same feature space, but have different data generating processes. While there is no limit on how much variance can exist in different but related domains, a common example in NLP is to treat different

* Corresponding author.
*E-mail address:* jworsha2@uccs.com (J. Worsham).

https://doi.org/10.1016/j.patrec.2020.05.031
0167-8655/

**Table 1**
TL / MTL Task and Domain Categories.

| | Same Tasks | Different Tasks |
|---|---|---|
| Same Domains | Standard Learning Setting | Inductive Transfer Learning / Inductive Bias (standard) MTL |
| Different Domains | Transductive Transfer Learning (Domain Adapation) / Transductive MTL (Domain Reuglarization) | Unsupervised Transfer Learning / Multi-Task Feature Learning |

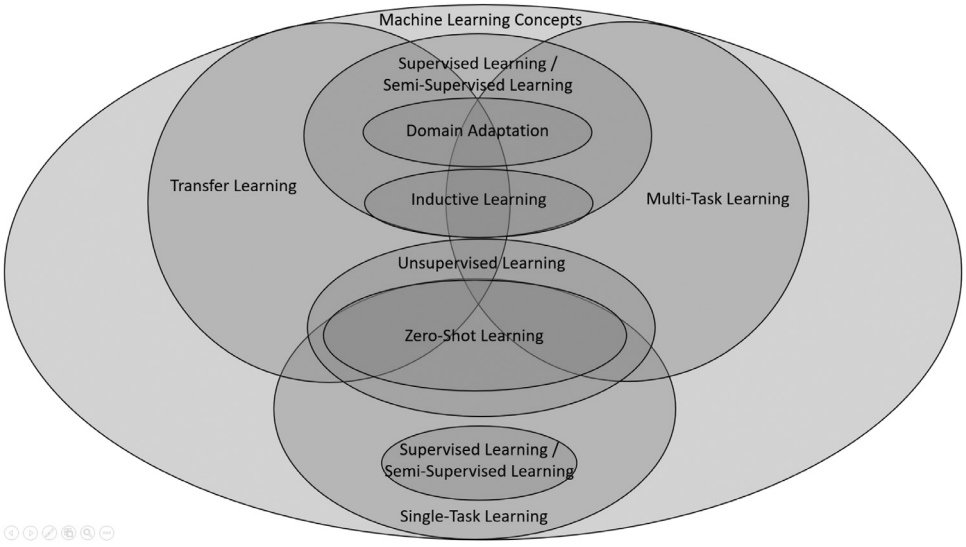


**Fig. 1.** Relationship of Machine Learning Concepts with a Focus on Transfer Learning and Multi-Task Learning.

languages as different domains [46]. Both TL and MTL can make use of differing domains and differing tasks.

Transfer Learning is broken down into three different categories based on what differs between the source and the target [32]. If the source and target share the same task with different domains, this is called Transductive Transfer Learning, commonly known as Domain Adaptation. If the source and target share the same domain with different tasks this is Inductive Transfer Learning which learns to transfer via inductive bias. If the source and target have different domains and different tasks this is a form of unsupervised Transfer Learning which learns common representations despite having no direct similarities between the source and the target.

With this TL taxonomy, we formulate a related breakdown for MTL. While MTL terminology is traditionally focused on varying tasks it is also possible to train jointly on different domains. If the source task and auxiliary tasks are the same with different domains, we label this Transductive Multi-Task Learning, or Domain Regularization. When the source task and auxiliary tasks are different but share the same domain this is the standard form of MTL which we formally identify as Inductive Bias MTL. Finally, if the source and auxiliary tasks are different and do not share the same domain we call it Multi-Task Feature Learning, originally introduced by Romera-Paredes et al. [33]. Table 1 shows this breakdown for both TL and MTL.

A relational representation of these concepts is provided in Fig. 1. This shows that TL and MTL share some overlap, indicating that these techniques can be used together. While Standley et al. [37] show that there are significant differences in the types of tasks proven to be useful in MTL vs. TL, Bingel and Søgaard [7] argue that they produce similar observed benefits. Liu et al. [20] show that TL and MTL are complementary to one another when used in combination to train a complex model, which is in contradiction to earlier work that showed, in different circumstances, this combination yielded no significant improvement [25]. These techniques also overlap with standard single-task learning through a method called zero-shot learning. Zero-shot learners, or generalist agents, are capable of jointly understanding many tasks or concepts with no fine-tuning on specific tasks [23].

## 3. When to use multi-task learning

One of the biggest needs for a successful machine learning system is access to extremely large amounts of labeled data. MTL is proposed as a technique to help overcome data sparsity by learning to jointly solve related or similar problems to produce a more generalized internal representation. Regardless of the number of target tasks to solve, MTL can only be considered useful when at least one target task is improved upon when compared to a collection of single-task models [37].

Along with enabling zero-shot learning [23], MTL is commonly presented as a regularization technique to aid in the generalization of a task to unseen examples [9,20,22,29]. This belongs to the Transductive MTL class in Table 1.

Additionally, MTL has desirable traits when it comes to efficiency. Stickland and Murray [38] describe a well designed MTL model to be computationally less complex, to have fewer parameters limiting the burden on memory and storage and to ultimately

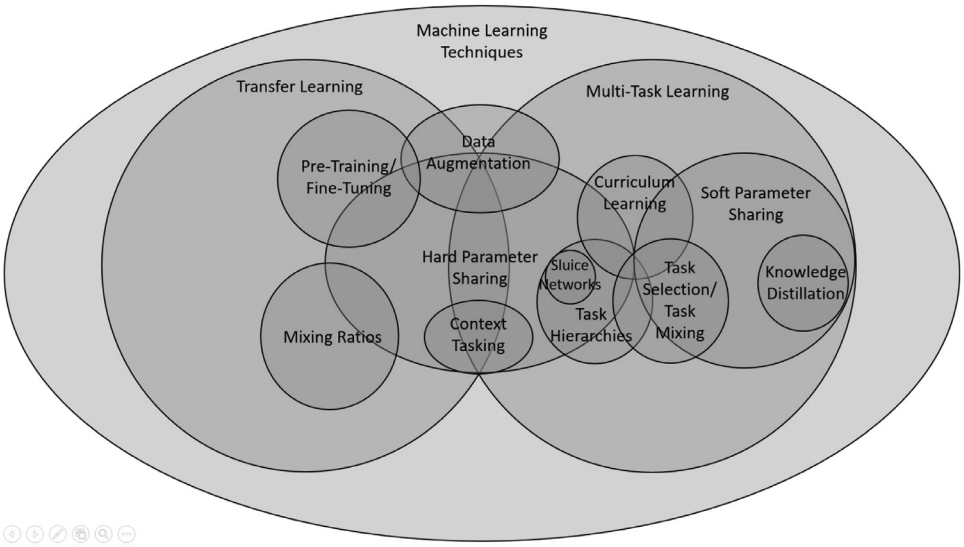


**Fig. 2.** Relationships of transfer learning and multi-task learning techniques for deep learning.

require less power consumption at inference time. Standley et al. [37] also point out the desirable trait of quicker inferencing depending on the architecture of the MTL model. These characteristics pertain to neural network MTL implementations which are less expensive than deploying a complete model for each class individually.

## 4. MTL implications and discoveries

Researchers have been studying the implications and nuances of MTL when compared to traditional single-task training since its introduction. Given the human intuition of how MTL can help improve model performance, practitioners are often surprised at how delicate and sensitive these algorithms can be [2,37]. This section will discuss MTL discoveries in this regard through the topics of task relationship, dataset diversity, model design considerations and training curriculum. Techniques identified in this section are shown relationally in Fig. 2.

### *4.1. Task selection*

The similarities between a set of tasks are commonly cited as one of the most influential design factors in building MTL systems. Through a series of experiments, Caruana [9] showed that the benefit of MTL is due to the direct knowledge learned from the auxiliary tasks. He further showed that some inductive bias can actually harm performance. A host of other researchers have gone on to argue that task relatedness plays a key role in determining how knowledge information is shared [6,25] and when an auxiliary task will help and when it will hurt [18,37].

Standley et al. [37] continued to explore this concept and showed that tasks which seem related can often have underlying complex and competing dynamics. They present a table showing how every factorized task pair performed in relation to a single-task model, and, via the hypothesis that related tasks improve performance, show which tasks they believe to be related. While this work was not performed on a set of NLP tasks, it showed the importance of task relationship and provided a novel way to measure relatedness.

Another unique observation published by Standley et al. [37] shows that tasks which are beneficial in a TL environment appear to perform poorly as auxiliary tasks in an MTL setting. This raises a question on whether or not dissimilar tasks could be used intentionally to help regularize a system or even protect a system from adversarial attack. While this observation seems to disagree with the conventional wisdom that only similar or related tasks can lead to improved model performance [6], they are not alone. Romera- Paredes et al. [33] have also shown that unrelated tasks can still be beneficial. This poses a unique opportunity to further explore task relationships and usefulness on the most recent MTL benchmarks.

### *4.2. MTL dataset considerations*

The datasets used for target tasks and auxiliary tasks play an important role in building successful MTL systems. The first topic addressed considers how the size and diversity of the datasets impact the learning of a model. Luong et al. [22] perform a set of experiments to determine how the size of the datasets for the target and auxiliary tasks impact the overall results of the model on the target set. They show that the size ratio between a target task dataset and an auxiliary task dataset does have an impact on performance. They argue that when the target dataset is large, the best MTL performance is achieved with a small auxiliary dataset with a size ratio between 0.01 and 0.1. When this mixing ratio gets too high it is shown that the model will overfit to the auxiliary task at the cost of performance on the target task. Other researchers agree that the best performance is achieved with a small number of auxiliary task updates compared to target task updates [2,18] and that adding more data to a poorly selected auxiliary task can significantly harm the model [18].

Researchers have also considered the underlying properties and statistics to determine how they impact MTL performance. A theoretical definition of MTL and task relatedness is presented by Ben-David and Borbely [6]. The goal of this work is to develop a formulated approach for determining when MTL is advantageous and to what degree. They seek a theoretical justification for task relatedness based on measurable similarities found within the underlying data generating processes of each task. While their definition of task more closely relates to the definition of a domain within this survey, they establish formal error bounds to measure and learn task relatedness.

Recent work has gone on to argue that size is not a useful metric for determining MTL gain [7]. Research has shown that simple tasks, requiring few training iterations, and difficult tasks, which struggle to converge on a solution, do not lead to the development

of useful MTL representations [9,23]. Alonso and Plank [2] argue that MTL task selection should be addressed via data properties, not intuition on what a human performer may consider easy. They perform a set of studies that measure statistical distributions of supervised labels in auxiliary task datasets and find that the best performance is achieved when the auxiliary tasks have compact mid-entropy distributions. That is to say, the best auxiliary tasks are neither too easy to predict nor too difficult to learn.

Another perspective on underlying properties of the auxiliary datasets is to consider the loss produced by each task while learning. The magnitude of the task loss can be considered a task similarity metric. Standley et al. [37] show that imbalanced tasks in this regard produce largely varied gradients which can confuse model training. Oftentimes, task selection is not something that can be changed, but Standley et al. [37] recommend using a task weighting coefficient to help normalize the gradient magnitudes across the tasks. Similar to task loss, the learning curve, showing how loss decreases over training, is also proposed as a metric for task similarity. It was found that MTL gains are more likely when a target task's learning curve plateaus early in training and the auxiliary tasks do not plateau [7].

It is also worth noting that Data Augmentation is a proven technique to help overcome data sparsity and improve task performance in TL and MTL settings. Anaby-Tavor et al. [3] propose LAMBADA, a language generator model fine-tuned on a small task-specific dataset, which generates semi-supervised training data to augment the data available to a task specific language classification task.

### 4.3. Model selection and design

There is a large body of research considering how MTL influences model selection and design. While the authors acknowledge the importance of other machine learning models, this section will focus solely on neural networks due to recent deep learning trends. Fig. 2 shows the primary MTL techniques for Deep Learning and their relationships. There are two primary families of MTL neural network approaches: hard parameter sharing and soft parameter sharing [34]. Hard parameter sharing is the approach most closely related to traditional machine learning techniques and the same mechanism used by many transfer learning solutions. In hard parameter sharing, full network layers and their parameters are shared directly between tasks. Soft parameter sharing is more specialized and instead creates separate layer parameters for each task. These task-specific layers are then regularized during training to reduce the differences between shared layers. This encourages layers to have similar weights but allows each task to specialize specific components. Diagrams showing these network concepts can be found in the MTL survey by Ruder [34]. Knowledge Distillation (KD) is a proven soft parameter technique which trains a student network to imitate the full output of a trained teacher network [15].

An early study in MTL showed that auxiliary tasks which help increase performance on a target task prefer to share hidden layers and weights with the target task, while unhelpful auxiliary tasks prefer to use weights not used by the target task [9]. This intuition has laid the groundwork for deep learning models which focus on building an enhanced internal representation of a problem space through shared hidden layers. It has been shown that predefining which layers to share can improve the performance of a deep learning MTL model when the tasks are generally beneficial, but this can break down if the wrong task pairs are selected [34]. The study goes on to argue for the benefit of learning task hierarchies internal to the model during training to help overcome this problem. Research has also shown that the depth of a layer and the benefit of sharing the layer between two tasks can be considered a measure of similarity of the two tasks [25]. They argue that low-level layers, such as word embeddings, are generally useful for all NLP tasks, while higher level layers become more specific and can only be shared among more similar tasks. This suggests that model architectures can be built off this metric when combined with other evaluations of task relatedness.

Another model consideration when building MTL systems is the capacity of the network. Radford et al. [29] prove that the capacity of a language model is essential to good performance and that increasing capacity produces a log-linear improvement. This follows conventional neural network wisdom and agrees with other research, such as BERT [12], whose performance appears to scale with the model size, and T5 [30], which achieved state-of-the-art results when pushed to 11 billion parameters.

Ruder et al. [35] show that hard-parameter sharing, task-specific network layers, hierarchical NLP layers and a regularizer to encourage tasks to share only what is useful, called block-sparse regularization, can be combined to create a powerful MTL network called a Sluice network. The Sluice network consistently performed better than single-task multi-layer perceptrons (MLPs) on all evaluation tasks and outperformed traditional hard parameter sharing approaches [9] on most NLP tasks.

An additional question that must be addressed is how a task is represented within the model. It is common with Inductive Bias MTL that each task has a specific set of output layers that can be queried to return task specific results [34]. However, McCann et al. [23] present a novel idea in which the task itself in included as input to the network, identified within this survey as Context Tasking. While the implementation may differ across domains and tasks, Context Tasking was implemented here by representing each task as a natural language question with a natural language answer. This avoids the need for any task-specialized components and naturally supports zero-shot learning and open-set classification [36]. Aralikatte et al. [4] present another interesting approach to Context Tasking by casting the NLP tasks of ellipsis resolution and coreference resolution as reading comprehension problems and produced new state-of-the-art results using Inductive Bias MTL.

### 4.4. Training curriculum

A final topic of MTL training implications is the design of a training curriculum. Given the research above regarding mixing ratios, task weighting, shared representations and sensitivities to task selection, it seems natural that MTL should be addressed with an intelligent training curriculum. The standard curriculum for MTL tasks is to build mini-batches containing examples for a single task and then alternate between tasks during training. The ratio of task mini-batches can be identical for all tasks or varied based on task performance or dataset size [9,22,34,43]. McCann et al. [23] refer to this as a fixed-order round robin curriculum and prove that it works well on tasks that require few iterations, but it struggles with more complex tasks. They furthermore consider hand-crafted curricula and show that beginning with more difficult tasks and slowly introducing additional tasks performs the best. Other work has considered including task-specific stopping conditions for TL and MTL [25], and more recent research has proposed a teacher-based annealing solution to dynamically control the auxiliary task impact with KD [10]. Other research has shown that round robin training is most impactful towards the end of the curriculum [38]. They propose a technique called annealed sampling in which batch sampling is originally based on the ratio of dataset sizes and slowly anneals to an even distribution across all tasks as the current epoch number increases. These discoveries, when combined with curriculum research emerging from the field of re-

inforcement learning [39], lead to a wealth of new research opportunities towards the design of MTL curricula.

## 5. Learning task relationships

Beyond the research into measuring properties of tasks and datasets to determine similarities, there have been multiple efforts to intrinsically learn task relatedness through a learning process. Caruana [9] showed that neural networks trained in an MTL setting exhibited a behavior where related tasks would share hidden nodes and unrelated tasks would not. This discovery implies that neural networks are able to determine what information is useful for sharing between tasks without an explicit signal conveying the task relationship. It is therefore reasonable to believe that neural networks are able to learn, and even describe, task relationship explicitly through the MTL training process. Research has since explored different clustering techniques built on this discovery which attempt to cluster network weights and parameters leading to a latent task relationship embedded in the task clusters [34]. Not only do these techniques inherently learn task relationships, they also help to train neural networks by penalizing them from diverging too much from a common set of knowledge shared by similar tasks. Ruder [34] also presents the Deep Relationship Network and the Cross-Stitch Network which are hard and soft parameter models, respectively, able to identify task relationship through training.

An approach called task2vec has been proposed which learns an embedding vector for an entire task that is agnostic to the size of the dataset [1]. The embedding attempts to capture semantic similarities between tasks by training a model to solve a task, and then probing the network to approximate the amount of information carried by the weights. The proximities between two task embedding vectors are theorized to represent task relatedness while the magnitude of the embedding vector is thought to correlate to the complexity of the task.

An alternative approach to learning directly from the hidden nodes and gradients is to efficiently search through task pairs to determine task similarities. Depending on the number of tasks an exhaustive search very quickly becomes impossible, however, heuristic based searches have been found to act as a good stand-in to estimate when tasks may be related [37]. They show that there is a high correlation between the validation loss of a network trained on 20% of the data and the fully trained validation loss of a network. Based on this claim, they use the loss at 20% as a heuristic to lightly train multi-task permutations for finding optimally performing task sets. They go on to show that given three tasks, the average loss of every two-pair combination is an effective approximation of the loss when all three tasks are trained jointly. This acts as a good search heuristic for finding optimized task sets. While this work has focused on small task sets and relatively small combinations, others have shown the benefit of having many auxiliary tasks to boost MTL performance [20,31,34]. More research into the implications of these findings is important to understanding the effect of the number of tasks present in an auxiliary task set.

## 6. MTL benchmarks and leaderboards

While there are many research efforts that evaluate MTL model performance on custom task sets, there exist several gold standard benchmarks which enable comparative evaluation. The first of these is the NLP Decathlon [23], decaNLP, which combines ten common NLP tasks/datasets: Question Answering, Machine Translation, Summarization, Natural Language Inference, Sentiment Analysis, Semantic Role Labeling, Zero-Shot Relation Extraction, Goal-Oriented Dialog, Semantic Parsing and Pronoun Resolution. Each task is assigned a scoring metric between 0 and 100. An overall decaScore is computed as the sum of all the task scores with the highest possible being 1000. Using the Context Tasking technique, every task is represented as a natural language question, a context and an answer. The decaNLP leaderboard presents an opportunity for MTL researchers to assess model performance.

One of the most popular evaluation benchmarks used for TL and MTL alike is the General Language Understanding Evaluation (GLUE) [43]. GLUE challenges models to solve the following 9 NLP tasks: Grammatical Acceptance Prediction, Sentiment Analysis, Paraphrasing, Semantic Equivalence, Semantic Similarity, Question Answering, Pronoun Resolution and two different Textual Entailment tasks. Each task has a defined scoring metric to evaluate task-specific performance; F1 score is commonly used among the tasks. GLUE does not require that all tasks be solved by the same model, and, as such, many top solutions have a fine-tuned model per task. An overall GLUE score, the macro-average of all tasks, is computed for each model.

In order to keep challenging researchers to push the state-of-the-art, an additional NLP benchmark, called SuperGLUE, is presented which is designed to be significantly more difficult [42]. The following 7 tasks are included in the SuperGLUE: Binary Question Answering, Imbalanced 3-Class Textual Entailment, Logical Causal Relationship, Textual Entailment, Binary Word-Sense Disambiguation, Pronoun Resolution and two different Multiple-Choice Question Answering tasks. Textual Entailment and Pronoun Resolution are the only two tasks from the original GLUE benchmark retained in SuperGLUE. These tasks were kept because they still showed room for improvement and proved to be two of the hardest tasks in GLUE.

## 7. MTL solutions for NLP

There is a rich library of research presenting technical implementations and use cases for MTL models and architectures. This section provides an overview of recent state-of-the-art approaches. Table 2 shows a comparison of model sizes and scores on common benchmarks.

### 7.1. Multi-task question answering network

The Multi-task Question Answering Network (MQAN) [23] is a natural language Context Tasking network designed to jointly learn over all tasks with no task specific weights or parameters in the network. All inputs and tasks are modeled as natural language questions and outputs in the form of a natural language answer. This enables the network to learn to solve tasks which traditionally have different input and output structures, such as machine translation and relation extraction. The authors show that MQAN is able to achieve performance comparable to ten single-task networks with no fine-tuning or task specific layers. Due to the common contextualized input design, MQAN is able to do zero-shot training and can even adapt to unseen classes in classification.

**Table 2**
MTL model comparison.

| Model | Params | GLUE | SuperGLUE | decaScore |
|---|---|---|---|---|
| MQAN | 29M | - | - | 609.0 |
| MT-DNN | 350M | 87.6 | - | - |
| $BERT_{Base}$ | 110M | 78.3 | - | - |
| $BERT_{Large}$ | 340M | 80.5 | 69.0 | - |
| BERT with PALs | 125M | - | - | - |
| BERT+BAM | 335M | 82.3 | - | - |
| RoBERTa | 375M | 88.5 | 84.6 | - |
| $ALBERT_{xxl}$ Ensemble | 235M | 89.4 | - | - |
| GPT-2 | 1,542M | - | - | - |
| XLNet-Large | 340M | 88.4 | - | - |
| T5-11B | 11B | 89.7 | 89.3 | - |

### 7.2. BERT and related models

Arguably one of the most important models recently proposed is BERT: Bidirectional Encoder Representations from Transformers [12]. BERT pre-trains a transformer model [40] with an unsupervised multi-task objective. This pre-training objective trains the network to predict a random mask of hidden words in a text document and to predict if a shown sentence is the logical next sentence in the document via a binary classifier. Along with the novel pre-training objective, BERT also presents a mechanism for contextualizing on both the left and right text directions while other popular models, such as GPT [28], are unidirectional. BERT scored competitively on the GLUE leaderboard and provided a base for researchers to build upon.

Since the release of BERT, there have been a number of modifications which have surpassed the baseline score on GLUE [17,19,27]. BERT and PALs train a single BERT model to be used for all tasks jointly, as opposed to building a fine-tuned model for each task [38]. Clark et al. [10] approach MTL with BERT from a different angle by doing multi-task fine-tuning through knowledge distillation and a multi-teacher paradigm, called BAM. The model is trained with a teacher annealing curriculum that gradually transfers the target learner from distillation through the teachers to a supervised MTL signal. RoBERTa is an optimized take on BERT that finds techniques which significantly improve performance [21]. ALBERT replaces the next sentence prediction task, proven ineffective by Yang et al. [45], with a sentence-order prediction pre-training task [19].

One notable extension of BERT with true multi-task learning across all 9 GLUE tasks is the Multi-Task Deep Neural Network (MT-DNN) [20]. The authors argue that MT-DNN has better domain transfer across tasks than standard BERT. The process begins with the regular BERT pre-training, followed by multi-task training with hard parameter sharing and a random round robin curriculum and finally ends with task-specific fine-tuning.

### 7.3. GPT/GPT-2

BERT is not the only type of language model that has successfully performed in MTL environments. GPT [28] is based on a multi-layer transformer network and GPT-2 [29] extends this model with an unsupervised multi-task pre-training objective. In inference settings GPT-2 is first task-conditioned to solve the desired task. This zero-shot type of learning can outperform the current state-of-the-art on a majority of NLP tasks. GPT-2 is also shown to perform competitively when used in a traditional pre-training and fine-tuning process. The authors have indicated that in future work they plan to assess GPT-2 performance on decaNLP and GLUE benchmarks.

### 7.4. XLNet

XLNet is proposed as a next-generation model which is intended to leverage the best features found in BERT and GPT while overcoming their intrinsic shortcomings [45]. The authors claim that BERT suffers from a pre-train/fine-tune discrepancy due to the masked words introduced in pre-training. While the masked words are helpful for building a latent understanding of language, masked words are never seen in practice and thus there is a distinct difference in the training data and real-world inputs. While this simplification has worked well for BERT, Yang et al. [45] attempt to improve performance by estimating the joint probability of the words seen in a piece of text. The authors also empirically show that BERT's next sentence prediction pre-training objective did not improve model performance and, hence, was dropped from the XLNet pre-training regimen.

### 7.5. T5

T5 [30] (Text-to-Text Transfer Transformer) is a refinement to the traditional transformer which boasts an unsupervised pre-training corpus of roughly 750 GB and uses natural language Context Tasking. The highest performing model designed by the authors contains 11 billion parameters, far more than what any other model has considered, and has beaten all other models addressed above on the GLUE and SuperGLUE leaderboards. This work provides convincing evidence regarding the claim that model capacity is an important factor in transfer learning and MTL in NLP.

## 8. Current challenges and opportunities

Most challenges that are still faced today in MTL are the same challenges that have existed for the past two decades. Caruana [9] proved that some inductive bias can hurt, and while it is still generally believed that task relatedness leads to good bias, there is no strong general notion of measuring this [6,34]. Standley et al. [37] begin to address this by confronting the underlying challenge of crosstalk, in which MTL suffers from complex and competing objectives. Additional studies have researched task relationship and performance on earlier model generations, such as bi-LSTMs [2,7,22]. Studies applying similar in-depth analysis to the most recent multi-task benchmarks with the latest transformer-based models are prime research opportunities to understand better the tasks to solve and the implications of the selected models.

Standley et al. [37] present several interesting claims which are worth exploring and applying to known MTL benchmarks. The first is that it could be better to train dissimilar tasks as opposed to semantically similar tasks. Additionally they argue that MTL performance estimates can be made by averaging the results of lesser-order task pairs. Both claims present research opportunities that could lead to better understanding of the impact of auxiliary task selection. The new set of MTL deep learning models should also be explored through probing in a manner similar to that of Kim et al. [18] to understand better the impact of NLP task selection. There still is a need for deeper and more general techniques for task selection and task assessment. As research dives deeper into the implications of MTL it is important to continue strengthening the current understanding of task relationship and selection.

Curriculum learning is continuing to gain popularity and will likely become of larger interest with the introduction of standardized MTL benchmarks. Curriculum learning has not been explored much in NLP or MTL, however, it has a rich history in reinforcement learning (RL) where curriculum is used to guide trained agents to more complex and realistic behaviors [13,39]. The curriculum is often generated in RL settings and it would be interesting to expand on these capabilities for MTL curriculum generation. These generations could leverage some form of relatedness [20] or be driven by unsupervised or latent signals [1]. Other research into lifelong learning and continuous learning [24,26] present new ideas and paradigms which are related to MTL and can be utilized to help solve the MTL tasks mentioned in this survey.

Although many unsupervised natural language understanding tasks have recently been used in a pre-training setting, Luong et al. [22] pose the question of how unsupervised objectives may impact MTL performance as auxiliary tasks. Building off the TL process there are open questions on how an MTL model can leverage the same unsupervised datasets. They argue that an auxiliary task must be compatible with the target task, and both intrinsic (perplexity) and extrinsic (accuracy) metrics must be improved on the target task when trained with the auxiliary task. Alonso and Plank [2] pose an additional question: most auxiliary tasks are classification tasks, how do regression tasks fare as auxiliary tasks? González-Garduno and Søgaard [14] provide an example of

this with text readability prediction and an auxiliary gaze prediction task. They showed that they only needed small samples from the auxiliary task, the selection of the auxiliary task was robust to small changes in the domain and the shared feature representation provably enhanced model performance. This work shows that further research into regression auxiliary tasks could help to advance MTL state-of-the-art. Finally, Liu et al. [20] present a unique opportunity to study how MTL architectures perform against adversarial tasks which could potentially lead to a new set of hardened auxiliary tasks. We hypothesize that Domain Regularization or Multi-Task Feature Learning could help machine learning models better withstand adversarial attacks.

Most recent advancements in TL and MTL are based off hard parameter sharing. How do model architectures, such as the transformer, perform when regularized with an MTL-based soft parameter sharing? How would this compare to standard models such as BERT and GPT and what other techniques can be borrowed from Ruder [34] for the latest generation of deep learning models?

Lastly, the biggest challenge faced in current MTL research is that fine-tuned single-task models consistently outperform non-fine-tuned MTL models that share layers [10,23]. MTL pre-training followed by single-task fine-tuning is able to leverage the rich knowledge acquired through inductive bias, but the impact of the strong supervised signal creates narrow experts which are able to outperform the generalized experts produced by MTL. While this is fine for narrow systems designed to solve problems with expansive training datasets, this gap needs to be closed to improve performance on data sparse tasks and domains. A long-term goal that will continue to persist is to develop general experts which can compete with their single-task counterparts [11].

Ultimately we find this ambitious task before us. To find ways to build robust and capable MTL models and help to enable the next generation of general Artificial Intelligence.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References


[1] A. Achille, M. Lam, R. Tewari, A. Ravichandran, S. Maji, C. Fowlkes, S. Soatto, P. Perona, Task2vec: task embedding for meta-learning, arXiv preprint arXiv:1902.03545 (2019).

[2] H.M. Alonso, B. Plank, When is multitask learning effective? Semantic sequence prediction under varying data conditions, in: EACL Vol. 1, 2017, pp. 44–53.

[3] A. Anaby-Tavor, B. Carmeli, E. Goldbraich, A. Kantor, G. Kour, S. Shlomov, N. Tepper, N. Zwerdling, Not enough data? Deep learning to the rescue!, arXiv preprint arXiv:1911.03118 (2019).

[4] R. Aralikatte, M. Lamm, D. Hardt, A. Søgaard, Ellipsis and coreference resolution as question answering, arXiv preprint arXiv:1908.11141 (2019).

[5] J. Baxter, Theoretical models of learning to learn, in: Learning to learn, Springer, 1998, pp. 71–94.

[6] S. Ben-David, R.S. Borbely, A notion of task relatedness yielding provable multiple-task learning guarantees, Mach. Learn. 73 (3) (2008) 273–287.

[7] J. Bingel, A. Søgaard, Identifying beneficial task relations for multi-task learning in deep neural networks, in: EACL Vol. 2, 2017, pp. 164–169.

[8] E.V. Bonilla, K.M. Chai, C. Williams, Multi-task gaussian process prediction, in: NIPS, 2008, pp. 153–160.

[9] R. Caruana, Multitask learning, Mach. Learn. 28 (1) (1997) 41–75.

[10] K. Clark, M.-T. Luong, U. Khandelwal, C.D. Manning, Q. Le, Bam! born-again multi-task networks for natural language understanding, in: ACL, 2019, pp. 5931–5937.

[11] J. Clune, Ai-gas: Ai-generating algorithms, an alternate paradigm for producing general artificial intelligence, arXiv preprint arXiv:1905.10985 (2019).

[12] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: NAACL-HLT Vol 1, 2019, pp. 4171–4186.

[13] C. Florensa, D. Held, M. Wulfmeier, M. Zhang, P. Abbeel, Reverse curriculum generation for reinforcement learning, arXiv preprint arXiv:1707.05300 (2017).

[14] A.V. González-Garduno, A. Søgaard, Learning to predict readability using eye–movement data from natives and learners, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[15] G. Hinton, O. Vinyals, J. Dean, Distilling the knowledge in a neural network, arXiv preprint arXiv:1503.02531 (2015).

[16] T. Jebara, Multi-task feature and kernel selection for svms, in: ICML, 2004, p. 55.

[17] M. Joshi, D. Chen, Y. Liu, D.S. Weld, L. Zettlemoyer, O. Levy, Spanbert: Improving pre-training by representing and predicting spans, arXiv preprint arXiv:1907.10529 (2019).

[18] N. Kim, R. Patel, A. Poliak, P. Xia, A. Wang, T. McCoy, I. Tenney, A. Ross, T. Linzen, B. Van Durme, et al., Probing what different nlp tasks teach machines about function word comprehension, in: SEM, 2019, pp. 235–249.

[19] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, R. Soricut, Albert: A lite bert for self-supervised learning of language representations, arXiv preprint arXiv:1909.11942 (2019).

[20] X. Liu, P. He, W. Chen, J. Gao, Multi-task deep neural networks for natural language understanding, arXiv preprint arXiv:1901.11504 (2019a).

[21] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, Roberta: a robustly optimized bert pretraining approach, arXiv preprint arXiv:1907.11692 (2019b).

[22] M.-T. Luong, Q.V. Le, I. Sutskever, O. Vinyals, L. Kaiser, Multi-task sequence to sequence learning, arXiv preprint arXiv:1511.06114 (2015).

[23] B. McCann, N.S. Keskar, C. Xiong, R. Socher, The natural language decathlon: Multitask learning as question answering, arXiv preprint arXiv:1806.08730 (2018).

[24] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, B. Yang, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, et al., Never-ending learning, Commun. ACM 61 (5) (2018) 103–115.

[25] L. Mou, Z. Meng, R. Yan, G. Li, Y. Xu, L. Zhang, Z. Jin, How transferable are neural networks in nlp applications? in: EMNLP, 2016, pp. 479–489.

[26] G.I. Parisi, R. Kemker, J.L. Part, C. Kanan, S. Wermter, Continual lifelong learning with neural networks: areview, Neural Networks (2019).

[27] J. Phang, T. Févry, S.R. Bowman, Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks, arXiv preprint arXiv:1811.01088 (2018).

[28] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, Improving language understanding by generative pre-training (2018).

[29] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language models are unsupervised multitask learners(2019).

[30] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P.J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, arXiv preprint arXiv:1910.10683 (2019).

[31] A. Ratner, B. Hancock, J. Dunnmon, R. Goldman, C. Ré, Snorkel metal: Weak supervision for multi-task learning, in: Workshop on Data Management for End–To-End Machine Learning, ACM, 2018, p. 3.

[32] I. Redko, E. Morvant, A. Habrard, M. Sebban, Y. Bennani, Advances in Domain Adaptation Theory, Elsevier, 2019.

[33] B. Romera-Paredes, A. Argyriou, N. Berthouze, M. Pontil, Exploiting unrelated tasks in multi-task learning, in: International Conference on Artificial Intelligence and Statistics, 2012, pp. 951–959.

[34] S. Ruder, An overview of multi-task learning in deep neural networks, arXiv preprint arXiv:1706.05098 (2017).

[35] S. Ruder, J. Bingel, I. Augenstein, A. Søgaard, Sluice networks: learning what to share between loosely related tasks, Stat 1050 (2017) 23.

[36] W.J. Scheirer, L.P. Jain, T.E. Boult, Probability models for open set recognition, IEEE Trans. Pattern Anal. Mach. Intell. 36 (11) (2014) 2317–2324.

[37] T. Standley, A.R. Zamir, D. Chen, L. Guibas, J. Malik, S. Savarese, Which tasks should be learned together in multi-task learning?, arXiv preprint arXiv:1905.07553 (2019).

[38] A.C. Stickland, I. Murray, Bert and pals: Projected attention layers for efficient adaptation in multi-task learning, in: ICML, 2019, pp. 5986–5995.

[39] M. Svetlik, M. Leonetti, J. Sinapov, R. Shah, N. Walker, P. Stone, Automatic curriculum graph generation for reinforcement learning agents, AAAI, 2017.

[40] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: NIPS, 2017, pp. 5998–6008.

[41] M. Velay, F. Daniel, Seq2seq and multi-task learning for joint intent and content extraction for domain specific interpreters, arXiv preprint arXiv:1808.00423 (2018).

[42] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, S.R. Bowman, Superglue: a stickier benchmark for general-purpose language understanding systems, arXiv preprint arXiv:1905.00537 (2019).

[43] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, S. Bowman, Glue: a multi–task benchmark and analysis platform for natural language understanding, in: EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, 2018, pp. 353–355.

[44] Q. Wang, L. Zhang, M. Chi, J. Guo, Mtforest: Ensemble decision trees based on multi-task learning., in: ECAI, volume 2008, 2008, pp. 122–126.

[45] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, Q.V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, arXiv preprint arXiv:1906.08237 (2019).

[46] B. Zoph, D. Yuret, J. May, K. Knight, Transfer learning for low-resource neural machine translation, arXiv preprint arXiv:1604.02201(2016).