Analysis and Prediction of NLP models via Task Embeddings

Anonymous ACL Rolling Review submission

Abstract

Relatedness between tasks, which is key to transfer learning, is often characterized by measuring the influence of tasks on one another during sequential or simultaneous training, with tasks being treated as black boxes. In this paper, we propose MetaEval, a set of 101 NLP tasks. We fit a single transformer to all MetaEval tasks jointly while conditioning it on low-dimensional task embeddings. The resulting task embeddings enable a novel analysis of the relatedness among tasks. We also show that task aspects can be used to predict task embeddings for new tasks without using any annotated examples. Predicted embeddings can modulate the encoder for zero-shot inference and outperform a zero-shot baseline on GLUE tasks. The provided multitask setup can function as a benchmark for future transfer learning research.

1 Introduction

Knowledge transfer from pretrained models has recently undergone considerable progress in NLP. Transformer-based encoders, such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2020), have achieved state-of-the-art results on text classification tasks. These models acquire rich text representations through masked language modeling (MLM) pretraining (Tenney et al., 2019; Warstadt et al., 2019, 2020b). However, these representations need additional task supervision to be useful for downstream tasks (Reimers and Gurevych, 2019). The default technique, *full fine-tuning*, optimizes all encoder weights alongside the training of the task-specific classifier.

The resulting encoder weights can be seen as a very high-dimensional[1] continuous representation of a model that is dedicated to a task $\mathcal{T}_i$ (Aghajanyan et al., 2020).

Continuous representations of tasks provide direct ways to probe the content of tasks and to assess the relationships among tasks. However, these possibilities are hindered by the very high dimensionality of the model weights. As a result, previous work on transfer learning treats each task as a black box instead of using continuous task representations. When tasks are viewed as black boxes, the task relationships are modeled by the influence they have on each other during sequential or joint training. For instance, Phang et al. (2018) note that fine-tuning on $\mathcal{T}_1$ = MNLI[2] and then on $\mathcal{T}_2$ = RTE (Dagan et al., 2006) outperforms directly fine-tuning on $\mathcal{T}_2$ = RTE. These findings lead to accuracy improvement but provide only coarse, unidimensional relations between tasks, and measuring all possible interactions among many tasks is computationally expensive.

Recently, Houlsby et al. (2019) proposed *adapters fine-tuning*, a strategy that considerably reduces the number of task-specific weights needed to achieve a performance comparable to full fine-tuning.[3] To do so, they freeze the pretrained transformer weights and insert residual, low-dimensional trainable modules A_{α_i} called adapters between transformer layers. During fine-tuning, each task can then be represented by α_i, the parameters of the adapters. Pilault et al. (2021) then showed that in a multitask setting with a collection of tasks Θ, a set of adapters $\{A_i, \mathcal{T}_i \in \Theta\}$ can be decomposed into two components: a set of task embeddings $\{z_i, \mathcal{T}_i \in \Theta\}$ and a single shared conditional adapter $A_\alpha(z_i)$. The task embeddings are trained jointly with the conditional adapter, which allows each task to modulate the

[1] E.g., $\approx 110M$ dimensions for $\text{BERT}_{\text{BASE}}$ full fine-tuning.

[2] MNLI (Williams et al., 2018) and RTE are two natural language inference (NLI) datasets. We call each dataset a task, even if they handle the same type of task, i.e., NLI.

[3] Houlsby et al. (2019) fine-tune the equivalent of 3% of BERT weights with a 0.4% GLUE (Wang et al., 2019b) average accuracy decrease compared to full fine-tuning

shared model in its own way. This approach leads to a performance improvement over individual adapters. Moreover, the parametrization is a very low-dimensional (dim(z) $\approx$ 100) task representation.

In this work, we leverage conditional adapters and propose a novel use of the obtained low-dimensional task embeddings. We derive task embeddings for 101 tasks based on a joint multitask training objective. This approach enables new analyses of the relationships among the tasks. Moreover, we show that we can predict the task embeddings from selected task aspects, which enables control of the model through the task aspects, thus contributing to selective and interpretable transfer learning.

We answer the following research questions: RQ1: How consistent is the structure of task embeddings? What is the importance of weight initialization randomness and sampling order on a task embedding position within a joint training run? How similar are task relationships across runs? RQ2: A consistent structure allows meaningful probing of the content of task embeddings. How well can we predict aspects of a task, such as the domain, the task type, or the dataset size, based on the task embedding? RQ3: Task embeddings can be predicted from task aspects, and a task embedding modulates a model. Consequently, can we predict an accurate model for zero-shot transfer based solely on the aspects of a task?

Since we study task representations, many tasks and, ideally, many instances for each task type are required for our analysis. Consequently, we have assembled 101 tasks in a benchmark that can be used for future probing and transfer learning. Our contributions are the following: (i) We assess low-dimensional task embeddings in novel ways, enabling their in-depth analysis; (ii) We show that these embeddings contribute to transferring models to target downstream NLP tasks even in situations where no annotated examples are available for training the downstream NLP task; (iii) We introduce MetaEval, a benchmark framework containing 101 NLP classification tasks.

2 Related Work

A common way to measure task relatedness is to train a model on a source task, or a combination of source tasks in the case of multitask learning (Caruana, 1997), and then measure the effect on

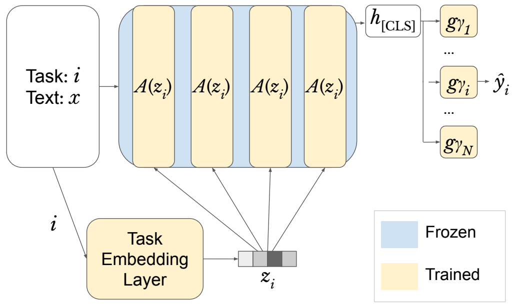

Figure 1: An overview of a transformer with a conditional adapter in a classification setup with N tasks. Batches for each task are used sequentially in random order. Each text example x is represented by $h_{[CLS]}$, which is the input of g_{γ_i} and the classifier for the task $\mathcal{T}_i$.

the target task's accuracy.

The search for the most useful source tasks for each target task has been the object of numerous studies. Mou et al. (2016) study the effect of transfer learning when the target task has a different domain from the source task and focus on different fine-tuning strategies, for instance, freezing or unfreezing specific layers. Conneau et al. (2017) train a sentence encoder with a selection of source tasks and show that natural language inference (NLI) provides the most transferable representations. Phang et al. (2018) also address the fine-tuning of pretrained BERT with a two-stage approach: an auxiliary pretraining stage on a source task before the final fine-tuning on the target task. Wu et al. (2020) investigate the phenomenon of negative transfer, i.e., the situation where source tasks harm target tasks in a multitask setting, and propose techniques to alleviate this phenomenon. D'Amour et al. (2020) show that when fine-tuning a model for a task, various random seeds can lead to similar accuracy but different behavior. We perform a similar analysis in a multitask setup and show that task embeddings are a valuable way to visualize this phenomenon.

By contrast, we do not study the influence of combinations of source tasks directly; we represent each task in a latent space. Our work is the first to evaluate the properties of tasks in the latent space and to predict task representations in that space instead of finding the most helpful source task.

Task embeddings in NLP have been introduced by Pilault et al. (2021). However, this work does not address analysis or prediction of the task embeddings but merely uses them as a proxy to ensure proper task coordination. Low-dimensional task representations have also been used as a way to measure the complexity of NLP tasks. Aghajanyan et al. (2020) show that ≈ 200 trainable parameters can guide random projections towards good approximations of full fine-tuning and use the number of trainable parameters required to achieve 90% of the full fine-tuning accuracy as a measure of task complexity. Continuous representation of a task has also been explored in computer vision by Achille et al. (2019), who interpret pooled Fisher information in convolutional neural networks as task embedding.[4] However, how to transpose this technique to a transformer architecture for use in NLP tasks is unclear.

Our work is also related to the probing of representations, which usually targets words (Nayak et al., 2016) or sentences. Conneau et al. (2018) probe sentence representations for various syntactical and surface aspects. Another type of probing, proposed for word embeddings, is the study of stability (Pierrejean and Tanguy, 2019; Antoniak and Mimno, 2018; Wendlandt et al., 2018). Stability measures the similarity of word neighborhoods across different training runs with varying random seeds.

3 Classification Models

We now introduce the classification models and fine-tuning techniques used in our experiments. To perform a classification task $\mathcal{T}_i$, we represent a text x (e.g., a sentence or a sentence pair) with an encoded [CLS] token $h_{[\text{CLS}]} = f_\theta(x)$. Here, f_θ is a transformer text encoder. $h_{[\text{CLS}]}$ is used as the input features for a classifier g. For each task, we use a different classification head g_{γ_i}, where γ_i represents softmax weights. To train a model for a task, we minimize the cross-entropy $\text{H}(y_i, g_{\gamma_i}(f_\theta(x))$.

Different strategies can be used to fine-tune a pretrained text encoder $f_{\theta_{\text{MLM}}}$ for a set of tasks.

Full Fine-Tuning is the optimization of all parameters of the transformer architecture alongside classifier weights, (θ_i, γ_i), independently for each task.

Adapters are lightweight modules with new parameters α that are inserted between each attention and feed-forward transformer layer (Houlsby et al., 2019). When using adapters (A_{α_i}), we freeze the transformer weights and represent each input text as $h_{[\text{CLS}]} = f_{\theta_{\text{MLM}}, A_{\alpha_i}}(x)$. During adapter fine-tuning, we optimize only the adapter weights and classifier weights (α_i, γ_i) for each task.

[4] Achille et al. (2019) work with classification and treat the detection of each label as a task. Fisher information is a way to measure the information carried by the convolutional filters for each label

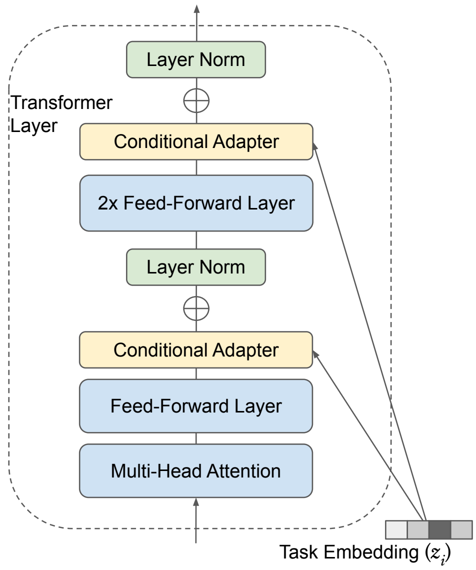

Figure 2: A transformer layer with conditional adapter layers.

Conditional Adapters We replace individual adapters with a conditional adapter $A_\alpha(z_i)$ that is common to all tasks but conditioned on task embeddings z_i.

Here, we train all the tasks jointly by optimizing a conditional adapter that learns to map each task embedding to a specific adaptation of the transformer weights while simultaneously optimizing the task embeddings. Figure 1 shows an overview of our conditional adapter setup. The objective is the following:

$$\min_{(\alpha, z_i, \gamma_i)} \sum_{\mathcal{T}_i \in \Theta} \text{H}(y_i, \hat{y}_i)$$

3.1 Parametrization of Adapters and Conditional Adapters

Figure 2 illustrates two conditional adapter layers in a transformer layer. An adapter layer is a perceptron with one hidden layer and a bottleneck of dimension d_A. Each adapter layer applies the following transformation:

$$h \to h + W_2 \sigma_A(W_1 h) \qquad (1)$$

where σ_A is an activation function and W_1, W_2 are projection matrices. Adapter layers[5] are then residually added between fixed-weight transformer layers to adjust the text representation for the target task.

Conditional adapters (Pilault et al., 2021) are an extension of adapters designed for parameter efficiency in multitask setups. For all tasks, a single conditional adapter is modulated by task-specific embeddings. When using conditional adapters, we first compute a d_A-dimensional gate:

$$\tau = \mathrm{sigmoid}(W_{\mathrm{gate}} z) \tag{2}$$

Then, we multiply by the hidden layer of the adapter.

$$h \rightarrow h + W_2 \sigma_A(\tau \odot W_1 h) \tag{3}$$

Each task embedding influences the gate, which in turn controls the activated dimensions of the conditional adapter. Tasks that are close in the task embedding space influence the feature extraction of the transformer in a similar way. Each layer has distinct conditional adapter weights, but a task embedding is shared across all layers.

4 Datasets

One of our goals is to study and leverage the task embeddings by making use of known task aspects. This process involves a mapping between the task and the aspects, which requires a varied set of tasks. The most commonly used evaluation suite, GLUE, contains only 8 datasets, which is not sufficient for our purpose. Therefore, we construct the largest set of NLP classification tasks[6] to date by casting them into the HuggingFace Datasets library.

HuggingFace Datasets (Wolf et al., 2020) is a repository containing individual tasks and benchmarks including GLUE (Wang et al., 2019b) and SuperGLUE (Wang et al., 2019a). We manually select classification tasks that can be performed from single-sentence or sentence-pair inputs and obtain 39 tasks.

CrowdFlower (Van Pelt and Sorokin, 2012) is a collection of datasets from the CrowdFlower platform for various tasks such as sentiment analysis, dialog act classification, stance classification, emotion classification, and audience prediction.

[5]Each layer has its own weights.

[6]We concentrate on English text classification tasks due to their widespread availability and standardized format.

Ethics (Hendrycks et al., 2021) is a set of ethical acceptability tasks containing natural language situation descriptions associated with acceptability judgment under 5 ethical frameworks.

PragmEval (Sileo et al., 2019) is a benchmark for language understanding that focuses on pragmatics and discourse-centered tasks containing 23 classification tasks.

Linguistic Probing (Conneau et al., 2018) is an evaluation designed to assess the ability of sentence embedding models to capture various linguistic properties of sentences with tasks focusing on sentence length, syntactic tree depth, word and part of speech content, and sensibility to word substitutions.

Recast (Poliak et al., 2018) reuses existing datasets and casts them as NLI tasks. For instance, an example in a pun detection dataset (Yang et al., 2015) *Masks have no face value* is converted to a labeled sentence pair (*Kim heard masks have no face value; Kim heard a pun* y=ENTAILMENT)

TweetEval (Barbieri et al., 2020) consists of classification tasks focused on tweets. The tasks include sentiment analysis, stance analysis, emotion detection, and emoji detection.

Blimp-Classification is a derivation of BLIMP (Warstadt et al., 2020a), a dataset of sentence pairs containing naturally occurring sentences and alterations of these sentences according to given linguistic phenomena. We recast this task as a classification task, where the original sentence is acceptable and the modified sentence is unacceptable.

The table in Appendix A displays an overview of the tasks in MetaEval. When splits are not available, we use 20% of the data as the test set and use the rest for an 80/20 training/validation split. We will make the datasets and splits publicly available.

5 Experiments

Our first goal is to analyze the structure and regularity of task embeddings. We then propose and evaluate a method to control models using task aspects.

5.1 Setup

Following Pilault et al. (2021), we use a RoBERTa$_{\mathrm{BASE}}$(Liu et al., 2020) pretrained trans-

Fine-Tuning Method	MetaEval Test Accuracy	Trained Encoder Parameters	Task Specific Trained Encoder Parameters
Majority Class	42.9	-	-
Full-Fine-Tuning (1 model/task)	76.9	124M	124M
Adapter	67.8	10M	10M
Conditional Adapter	**79.7**	10M	32

Table 1: Parameter counts and MetaEval test accuracy percentages of fine-tuning techniques.

former[7] with conditional adapters of size $d_A = 256$, a sequence length of 128, and Adam with a learning rate of 2.10^{-5} as an optimizer. We sample $30k$ training examples per task to limit the required computation time.

Multitask setup When multitasking, we sample one task from among all MetaEval tasks at each training step. We limit the loss of each task to 1.0, and sample each task at a rate proportional to the square root of the capped size (Stickland and Murray, 2019) to balance the mutual influence of the tasks. We use task embeddings of dimension 32, which was selected according to MetaEval average validation accuracy among $\{2, 8, 32, 128, 512\}$.

5.2 Target Task Results

We first evaluate the individual model performance for the settings described in section 3.

Table 1 compares the unweighted average of the accuracies computed for MetaEval tasks and the number of trainable parameters associated with the fine-tuning strategies. The conditional adapter model achieves comparable accuracy to that of full fine-tuning despite having only 32 task-specific encoder parameters per task. This ensures that task embeddings are accurate representations of tasks.

5.3 Geometry of Task Embeddings

Figure 3 displays a 2D projection of the task embeddings with UMAP (McInnes et al., 2018). Some task types, such as sentiment analysis and grammatical properties prediction, form distinct clusters. Moreover, a PCA projection, which is less readable but provides a more faithful depiction of the global structure, is shown in Appendix C.[8] This approach allows us to identify linguistic probing tasks (prediction of the number of objects/subjects, prediction of text length, prediction of constituent patterns) as outliers. Since the task embeddings reflect an influence on the conditional adapter, distance from the center can be seen as a way to measure task specificity. Tasks whose embeddings are far from the center need to activate the conditional adapter in a way that is not widely shared and are therefore more specific.

[7] $\text{BERT}_{\text{BASE}}$ had a similar behavior in our experiments, but with a slightly lower accuracy.

[8] Unlike UMAP, PCA is a *linear* projection of the original space.

Task Type	Position Stability
Grammar	62.0 ± 3.9
Acceptability	57.1 ± 0.0
Emotion	47.6 ± 2.2
Discourse	45.7 ± 0.0
NLI	37.5 ± 1.0
Other	34.8 ± 0.7
Paraphrase detection	31.5 ± 13.1
Facticity	30.0 ± 4.7
Random embedding	1.0 ± 0.5

Table 2: Task embeddings position stability within a training run according to task types. As a reference, we provide the expected stability that would be obtained for randomly sampled task embedding positions.

5.4 Stability Analysis

The appeal of task embeddings relies on the hypothesis that they form similar structures across runs and that each task has a position that does not depend excessively on randomness. In this section, we address these concerns.

5.4.1 Stability within a Run We investigate the sensitivity of task embeddings to initialization and to data sampling order by running the multitask training while assigning 3 embeddings with different initializations ($z_{i,1}, z_{i,2}, z_{i,3}$) to each task instead of 1. During training, one of the three embeddings is selected randomly in each task training step.

Figure 4 in Appendix B displays the task embedding space in this setting. Some task embeddings converge to nearly identical positions (*trec, rotten tomatoes, sst2, mnli*), while the embeddings of other tasks (*boolq, mrpc, answer_selection_experiments*) occupy a wider por-

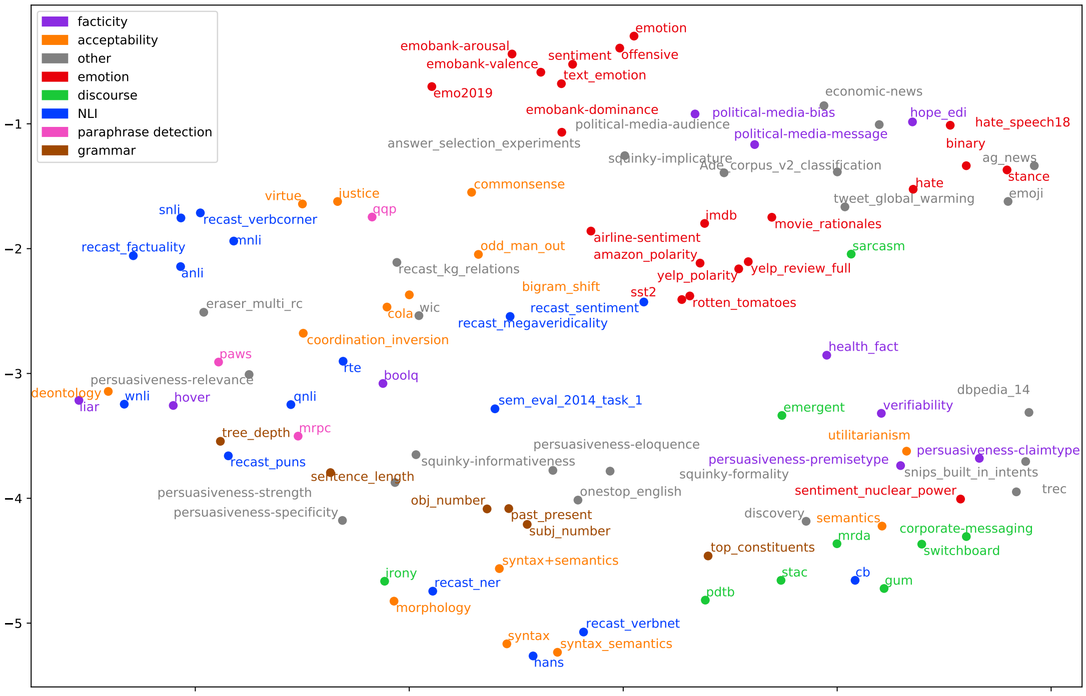

Figure 3: UMAP projection of the task embeddings.

Task Type	Neighborhood Stability
Emotion	26.3 ± 11.2
Grammar	20.2 ± 10.4
Acceptability	19.4 ± 9.1
Paraphrase detection	14.3 ± 10.4
NLI	14.1 ± 9.5
Facticity	13.1 ± 8.5
Discourse	11.6 ± 7.5
Other	10.2 ± 8.2

Table 3: Task embedding neighborhood stability according to task type.

tion of the embedding space. For each task, we compute the rate at which the 10 nearest neighbors[9] of an embedding $z_{i,k}$ contain an embedding of the same task with a different initialization, $z_{i,k'}, k' \neq k$.

The stability rates are reported in Table 2. The standard deviations (computed across runs) show that sensitivity to random seeds is inherent to the task groups. Some tasks occupy specific regions in the latent space, while other tasks can lie on multiple positions in a manifold. However, the variability is far from that of random positions.

5.4.2 Stability of Task Neighborhood

We study the neighborhood of each task embedding. Following Antoniak and Mimno (2018), we define the stability rate for a task embedding as the average overlap rate (according to the Jaccard metric) of the neighborhoods.

Given two spaces A and B from different runs and a task $\mathcal{T}_i$, we define the neighborhood of $\mathcal{T}_i$ in A as the top 10 closest other tasks according to cosine similarity. We also compute the neighborhood of $\mathcal{T}_i$ in B. We report the results according to task type in Table 3. The results show that the global structure of the space can change and that task type influences the neighborhood stability.

RQ1 can be answered with a distinction on the task type. The position of a task embedding within a run is relatively robust to randomness. Across runs, the organization of the task embedding space may vary. In both cases, lower-level tasks, such as grammar, acceptability, and emotion tasks, exhibit the most consistent structure.

5.5 Probing Task Embeddings for Task Aspects

We now use the task embeddings to investigate which task aspects influence the NLP models. Prior work developed a probing methodology to interpret the content of *text* embeddings. Conneau et al. (2018) selected an array of text aspects to see if they were contained in the text embedding. These aspects include text length, word content, the number of subjects and objects, the tense, natural word order, and syntactic properties.

To derive analogous *task* aspects $Lambda_i$, we

[9] According to cosine similarity.

Model	Domain-Cluster	Num-Examples	Num-Text-Fields	Task-Type	Text-Length
Majority Class	27.8	**62.3**	63.3	19.8	19.8
Logistic Regression	23.8	40.3	58.3	26.7	21.8
Gradient Boosting Classifier	34.8	45.7	**69.2**	29.8	29.0
KNN Classifer	**38.8**	35.8	68.2	**34.7**	**37.8**

Table 4: Accuracy of task aspect classification from task embeddings.

Model	All	Domain-Cluster	Num-Examples	Num-Text-Fields	Task-Type	Text-Length
Average Task Embedding	1.000	-	-	-	-	-
KNN Regression	0.955	0.986	1.056	1.000	0.953	1.021
Ridge Regression	0.907	0.956	0.997	1.005	0.918	0.988

Table 5: Mean squared error (MSE) of embedding regression from aspects. To normalize the reported MSE values, we divide them by the MSE of average task embedding prediction.

model a task as a collection of text examples with labels. We propose as aspects the number of text examples, the number of text fields per example, and the type of task. We also include basic properties derived from the text of the examples, namely, the median text length and the domain.

Num-Examples represents the number of training examples for a task. We discretize this value into 4 quartiles[10] computed across all tasks.

Num-Text-Fields is equal to 2 in sentence-pair classification tasks (e.g., NLI or paraphrase detection) and equal to 1 in single-sentence classification tasks (e.g., standard sentiment analysis).

Domain-Cluster is a representation of the domain of the input text of a task. Following (Sia et al., 2020), we represent the text of each task by the average spherical embedding (Meng et al., 2019). The domain of each task is represented by the average of the text embeddings of its examples. We then perform clustering across all task domains to reduce the dimensionality of the domain representation. We use Gaussian mixture model soft clustering and represent the domain by 8 cluster activations.[11]

Text-Length represents the length of the input examples (and the sum of input lengths when there are two inputs). We discretize this value into 4 quartiles computed across all tasks.

Task-Type is the type of task, selected from { ACCEPTABILITY, DISCOURSE, EMOTION, GRAMMAR, PARAPHRASE DETECTION, OTHER }.

Note that the above features do not rely on annotated data (only on the input text, sizes, and task type). We use logistic regression, a gradient boosting classifier, and a KNN classifier with Scikit-Learn (Pedregosa et al., 2011) default parameters[12] to learn to predict the aspects from task embeddings. Table 4 displays the classification accuracy for each aspect obtained by performing cross-validation with a leave-one-out split.

The number of training examples is limited to the number of tasks, which prevents high accuracy. However, our results address **RQ2** by showing that a simple linear probe can still capture the domain, the task type, and the length of the input. We could have expected a separation between classification of relationships between sentence pairs and single-sentence classification, but the task embeddings do not seem to accurately capture that aspect.

5.6 Task Embedding Regression

We now address the prediction of task embeddings from the previously defined aspects.

We use task embeddings z_i trained on the MetaEval multitask setup and then train a regression model to predict the task embeddings from the task aspects $\Lambda_{\mathcal{T}_i}$.

$$\hat{z}_i = \text{Regression}([a, a \in \Lambda_{\mathcal{T}_i}]) \quad (4)$$

We propose two evaluations, intrinsic and extrinsic. In our first evaluation, we directly measure the regression error, which allows us to measure how well trained task embeddings can be recovered on

[10] We experimented with finer quantizations, but they led to excessive sparsity.

[11] The number of clusters was selected with the elbow method.

[12] Release 0.24.1; deviation from the default parameters did not lead to a significant improvement.

	CoLa	SST2	MRPC	QQP	MNLI	QNLI	RTE	AVG
Single-Task Full-Fine-Tuning (Supervised)	79.2	93.1	75.5	84.7	80.9	88.9	47.3	78.5
Same Task-Type Full Fine-Tuning (ZS)	73.5	**93.6**	68.8	55.3	**72.7**	51.5	**70.2**	69.4
Aspect-Aware Task Embeddings (ZS)	75.4	90.0	**70.4**	**71.1**	66.2	**56.2**	63.7	**70.4**
Offline Task Embedding Ridge Regression (ZS)	76.2	92.0	67.6	61.6	71.7	53.8	68.6	70.2
Same Task-Type Task Embeddings (ZS)	**76.7**	91.4	67.6	57.0	67.0	53.8	64.0	68.2

Table 6: Zero-Shot (ZS) accuracy on GLUE tasks after training on MetaEval while excluding GLUE tasks (ME\G). As a reference, we also provide results with supervision on the evaluated task with the setup from section 5.1. The Same-Task-Type is the baseline, where for each task, RoBERTa is fine-tuned on (ME\G) same-type tasks while sharing label weights. The next methods use task embedding prediction via either offline or online regression, as described in section 5.6.

the basis of aspects alone. Table 5 shows the error with two regression models. The ridge regression model outperforms neighborhood-based regression (KNN), which shows that relevant aspects can be abstracted from the embeddings even on ≈ 100 examples.

In our second evaluation, we exclude GLUE tasks from MetaEval during the multitask conditional adapter training. We now share the label names across tasks during the multitask training to enable zero-shot inference. Then, we estimate task embeddings for the GLUE classification tasks from the aspects via logistic regression. We propose two different techniques for task embedding regression:

Offline Task Embedding Regression We first perform multitask training, then train a regression model to estimate task embeddings from a set of aspects. One advantage of this technique is that it allows the use of any aspect after multitask training. However, the model has to learn this relationship from only 100 examples since an example is a task.

Aspect-Aware Task Embeddings We propose another variation, where we perform multitask training and the regression of embeddings jointly. Instead of having a single task embedding z_i for each task $\mathcal{T}_i$, we augment it with an embedding z_{a_i} for each aspect a_i of $\mathcal{T}_i$. The task embedding modulating the adapters is then:

$$z_i + \sum_{a_i \in \Lambda_{\mathcal{T}_i}} z_{a_i} \quad (5)$$

An unseen task $\mathcal{T}_i$ can be represented by the sum of its aspect embeddings augmented with the average task embedding.

These two models use only the aspects of each GLUE task and not the annotated data.

As a baseline, we propose the **Same-Task-Type Full Fine-Tuning** of a RoBERTa model. For each GLUE task, we fine-tune the model on all MetaEval tasks of the same task type (Mou et al., 2016) while excluding GLUE tasks. For instance, to derive predictions on RTE, we fine-tune a RoBERTa model on all NLI tasks of MetaEval that are not in GLUE while sharing the labels. We also report the results of supervised RoBERTa models trained on each GLUE task with the hyperparameters described in section 5.1.

Table 6 reports the GLUE accuracy under both settings. Task embedding regression improves the average accuracy compared to that of the Same-Task-Type RoBERTa baseline. Learning aspect embeddings during multitask training leads to an improved average result, but most of the gain over the baseline can be achieved via offline regression. Finally, averaging the task embeddings of the same-type tasks leads to the worst results, which confirms the need to combine multiple aspects of a task for task embedding prediction. This finding addresses **RQ3** and establishes task embeddings as a viable gateway for zero-shot transfer.

6 Conclusion

We proposed a framework for the analysis and prediction of task embeddings in NLP. We showed that the task embedding space exhibits a consistent structure but that there are individual variations according to task type. Furthermore, we have demonstrated that task embeddings can be predicted based of the aspects of the tasks. Since the task embedding leads to a model, model manipulation can be performed according to desirable aspects for zero-shot prediction. Future work can consider new task aspects for model manipulation, for instance, the use of unwanted features or the language of the text.

References

Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless C Fowlkes, Stefano Soatto, and Pietro Perona. 2019. Task2vec: Task embedding for meta-learning. In *Proceedings of ICCV2019*, pages 6430–6439.

Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. 2020. Intrinsic Dimensionality Explains the Effectiveness of Language Model Fine-Tuning.

Maria Antoniak and David Mimno. 2018. Evaluating the stability of embedding-based word similarities. *Transactions of the Association for Computational Linguistics*, 6:107–119.

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa-Anke, and Leonardo Neves. 2020. Tweet-Eval:Unified Benchmark and Comparative Evaluation for Tweet Classification. In *Proceedings of Findings of EMNLP*.

Rich Caruana. 1997. Multitask Learning. *Machine Learning*, 28(1):41–75.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of EMNLP2017*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of ACL2018*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 177–190, Berlin, Heidelberg. Springer Berlin Heidelberg.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, and others. 2020. Underspecification presents challenges for credibility in modern machine learning. *arXiv preprint arXiv:2011.03395*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL2019*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. 2021. Aligning ai with shared human values. In *International Conference on Learning Representations*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-Efficient Transfer Learning for {NLP}. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799, Long Beach, California, USA. PMLR.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Ro{bert}a: A robustly optimized {bert} pretraining approach.

L. McInnes, J. Healy, and J. Melville. 2018. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv e-prints*.

Yu Meng, Jiaxin Huang, Guangyuan Wang, Chao Zhang, Honglei Zhuang, Lance Kaplan, and Jiawei Han. 2019. Spherical text embedding. In *Advances in neural information processing systems*.

Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. 2016. How transferable are neural networks in NLP applications? In *Proceedings of EMNLP2016*, pages 479–489, Austin, Texas. Association for Computational Linguistics.

Neha Nayak, Gabor Angeli, and Christopher D. Manning. 2016. Evaluating word embeddings using a representative suite of practical tasks. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 19–23, Berlin, Germany. Association for Computational Linguistics.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence Encoders on STILTs: Supplementary Training on Intermediate Labeled-data Tasks. *arXiv preprint arXiv:1811.01088v2*.

Bénédicte Pierrejean and Ludovic Tanguy. 2019. Investigating the stability of concrete nouns in word embeddings. In *Proceedings of the 13th International Conference on Computational Semantics - Short Papers*, pages 65–70, Gothenburg, Sweden. Association for Computational Linguistics.

Jonathan Pilault, Amine Elhattami, and Christopher Pal. 2021. Conditionally adaptive multi-task learning: Improving transfer learning in nlp using fewer parameters & less data. In *Submitted to International Conference on Learning Representations*. Under review.

Adam Poliak, Aparajita Haldar, Rachel Rudinger, J. Edward Hu, Ellie Pavlick, Aaron Steven White, and Benjamin Van Durme. 2018. Collecting diverse natural language inference problems for sentence representation evaluation. In *Proceedings EMNLP2018*, pages 67–81, Brussels, Belgium. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of EMNLP2019*. Association for Computational Linguistics.

Suzanna Sia, Ayush Dalmia, and Sabrina J. Mielke. 2020. Tired of topic models? clusters of pretrained word embeddings make for fast and good topics too! In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1728–1736, Online. Association for Computational Linguistics.

Damien Sileo, Tim Van de Cruys, Camille Pradel, and Philippe Muller. 2019. Discourse-based evaluation of language understanding.

Asa Cooper Stickland and Iain Murray. 2019. BERT and PALs: Projected attention layers for efficient adaptation in multi-task learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5986–5995, Long Beach, California, USA. PMLR.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of ACL2019*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Chris Van Pelt and Alex Sorokin. 2012. Designing a scalable crowdsourcing platform. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 765–766.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in neural information processing systems*, pages 3266–3280.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019b. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *International Conference on Learning Representations*.

Alex Warstadt, Yu Cao, Ioana Grosu, Wei Peng, Hagen Blix, Yining Nie, Anna Alsop, Shikha Bordia, Haokun Liu, Alicia Parrish, S Wang, Jason Phang, Anhad Mohananey, Phu Mon Htut, Paloma Jeretic, and Samuel R Bowman. 2019. Investigating BERT's Knowledge of Language: Five Analysis Methods with NPIs. In *EMNLP/IJCNLP*.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020a. BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Alex Warstadt, Yian Zhang, Xiaocheng Li, Haokun Liu, and Samuel R Bowman. 2020b. Learning Which Features Matter: RoBERTa Acquires a Preference for Linguistic Generalizations (Eventually). In *Proceedings of EMNLP2020*, pages 217–235, Online. Association for Computational Linguistics.

Laura Wendlandt, Jonathan K. Kummerfeld, and Rada Mihalcea. 2018. Factors influencing the surprising instability of word embeddings. In *Proceedings of NAACL2018*, pages 2092–2102, New Orleans, Louisiana. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of NAACL2018*, pages 1112–1122. Association for Computational Linguistics.

Thomas Wolf, Quentin Lhoest, Patrick von Platen, Yacine Jernite, Mariama Drame, Julien Plu, Julien Chaumond, Clement Delangue, Clara Ma, Abhishek Thakur, Suraj Patil, Joe Davison, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angie McMillan-Major, Simon Brandeis, Sylvain Gugger, François Lagunas, Lysandre Debut, Morgan Funtowicz, Anthony Moi, Sasha Rush, Philipp Schmidd, Pierric Cistac, Victor Muštar, Jeff Boudier, and Anna Tordjmann. 2020. Datasets. *GitHub. Note: https://github.com/huggingface/datasets*, 1.

Sen Wu, Hongyang R. Zhang, and Christopher Ré. 2020. Understanding and improving information transfer in multi-task learning. In *International Conference on Learning Representations*.

Diyi Yang, Alon Lavie, Chris Dyer, and Eduard Hovy. 2015. Humor recognition and humor anchor extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2367–2376, Lisbon, Portugal. Association for Computational Linguistics.

A List of Tasks

Dataset	Labels	Splits Sizes
health_fact/default	[false, mixture, true, unproven]	10k/1k/1k
ethics/commonsense	[acceptable, unacceptable]	14k/4k/4k
ethics/deontology	[acceptable, unacceptable]	18k/4k/4k
ethics/justice	[acceptable, unacceptable]	22k/3k/2k
ethics/utilitarianism	[acceptable, unacceptable]	14k/5k/4k
ethics/virtue	[acceptable, unacceptable]	28k/5k/5k
discovery/discovery	[[no-conn], absolutely,, accordingly, actually...	2M/87k/87k
ethos/binary	[no_hate_speech, hate_speech]	998
emotion/default	[sadness, joy, love, anger, fear, surprise]	16k/2k/2k
hate_speech18/default	[noHate, hate, idk/skip, relation]	11k
pragmeval/verifiability	[experiential, unverifiable, non-experiential]	6k/2k/634
pragmeval/emobank-arousal	[low, high]	5k/684/683
pragmeval/switchboard	[Response Acknowledgement, Uninterpretable, Or...	19k/2k/649
pragmeval/persuasiveness-eloquence	[low, high]	725/91/90
pragmeval/mrda	[Declarative-Question, Statement, Reject, Or-C...	14k/6k/2k
pragmeval/gum	[preparation, evaluation, circumstance, soluti...	2k/259/248
pragmeval/emergent	[observing, for, against]	2k/259/259
pragmeval/persuasiveness-relevance	[low, high]	725/91/90
pragmeval/persuasiveness-specificity	[low, high]	504/62/62
pragmeval/persuasiveness-strength	[low, high]	371/46/46
pragmeval/emobank-dominance	[low, high]	6k/798/798
pragmeval/squinky-implicature	[low, high]	4k/465/465
pragmeval/sarcasm	[notsarc, sarc]	4k/469/469
pragmeval/squinky-formality	[low, high]	4k/453/452
pragmeval/stac	[Comment, Contrast, Q_Elab, Parallel, Explanat...	11k/1k/1k
pragmeval/pdtb	[Synchrony, Contrast, Asynchronous, Conjunctio...	13k/1k/1k
pragmeval/persuasiveness-premisetype	[testimony, warrant, invented_instance, common...	566/71/70
pragmeval/squinky-informativeness	[low, high]	4k/465/464
pragmeval/persuasiveness-claimtype	[Value, Fact, Policy]	160/20/19
pragmeval/emobank-valence	[low, high]	5k/644/643
hope_edi/english	[Hope_speech, Non_hope_speech, not-English]	23k/3k
snli/plain_text	[entailment, neutral, contradiction]	550k/10k/10k
paws/labeled_final	[0, 1]	49k/8k/8k
imdb/plain_text	[neg, pos]	50k/25k/25k
crowdflower/sentiment_nuclear_power	[Neutral / author is just sharing information,...	190
crowdflower/tweet_global_warming	[Yes, No]	4k
crowdflower/airline-sentiment	[neutral, positive, negative]	15k
crowdflower/corporate-messaging	[Information, Action, Exclude, Dialogue]	3k
crowdflower/economic-news	[not sure, yes, no]	8k
crowdflower/political-media-audience	[constituency, national]	5k
crowdflower/political-media-bias	[partisan, neutral]	5k
crowdflower/political-media-message	[information, support, policy, constituency, p...	5k
crowdflower/text_emotion	[sadness, empty, relief, hate, worry, enthusia...	40k
emo/emo2019	[others, happy, sad, angry]	30k/6k
glue/cola	[unacceptable, acceptable]	9k/1k/1k
glue/sst2	[negative, positive]	67k/2k/872
glue/mrpc	[not_equivalent, equivalent]	4k/2k/408
glue/qqp	[not_duplicate, duplicate]	391k/364k/40k
glue/mnli	[entailment, neutral, contradiction]	393k/10k/10k
glue/qnli	[entailment, not_entailment]	105k/5k/5k
glue/rte	[entailment, not_entailment]	3k/2k/277
glue/wnli	[not_entailment, entailment]	635/146/71
glue/ax	[entailment, neutral, contradiction]	1k
yelp_review_full/yelp_review_full	[1 star, 2 star, 3 stars, 4 stars, 5 stars]	650k/50k
blimp_classification/syntax_semantics	[acceptable, unacceptable]	26k
blimp_classification/syntax+semantics	[acceptable, unacceptable]	2k
blimp_classification/morphology	[acceptable, unacceptable]	36k
blimp_classification/syntax	[acceptable, unacceptable]	52k
blimp_classification/semantics	[acceptable, unacceptable]	18k
recast/recast_kg_relations	[1, 2, 3, 4, 5, 6]	22k/2k/761
recast/recast_puns	[not-entailed, entailed]	14k/2k/2k
recast/recast_factuality	[not-entailed, entailed]	38k/5k/4k
recast/recast_verbnet	[not-entailed, entailed]	1k/160/143

Continued on next page

Dataset	Labels	Splits Sizes
recast/recast_verbcorner	[not-entailed, entailed]	111k/14k/14k
recast/recast_ner	[not-entailed, entailed]	124k/38k/36k
recast/recast_sentiment	[not-entailed, entailed]	5k/600/600
recast/recast_megaveridicality	[not-entailed, entailed]	9k/1k/1k
ag_news/default	[World, Sports, Business, Sci/Tech]	120k/8k
super_glue/boolq	[False, True]	9k/3k/3k
super_glue/cb	[entailment, contradiction, neutral]	250/250/56
super_glue/wic	[False, True]	5k/1k/638
super_glue/axb	[entailment, not_entailment]	1k
super_glue/axg	[entailment, not_entailment]	356
ade_corpus_v2/Ade_corpus_v2_classification	[Not-Related, Related]	24k
tweeteval/emoji	[_red_heart_, _smiling_face_with_hearteyes_, ...	50k/45k/5k
tweeteval/hate	[not-hate, hate]	9k/3k/1k
tweeteval/irony	[non_irony, irony]	3k/955/784
tweeteval/offensive	[not-offensive, offensive]	12k/1k/860
tweeteval/sentiment	[negative, neutral, positive]	46k/12k/2k
tweeteval/stance	[negative, neutral, positive]	3k/1k/294
trec/default	[manner, cremat, animal, exp, ind, gr, title, ...	5k/500
yelp_polarity/plain_text	[1, 2]	560k/38k
rotten_tomatoes/default	[neg, pos]	9k/1k/1k
anli/plain_text	[entailment, neutral, contradiction]	100k/45k/17k
liar/default	[false, half-true, mostly-true, true, barely-t...	10k/1k/1k
linguisticprobing/subj_number	[NN, NNS]	82k/8k/8k
linguisticprobing/obj_number	[NN, NNS]	80k/8k/8k
linguisticprobing/past_present	[PAST, PRES]	86k/9k/9k
linguisticprobing/sentence_length	[0, 1, 2, 3, 4, 5]	87k/9k/9k
linguisticprobing/top_constituents	[ADVP_NP_VP_., CC_ADVP_NP_VP_., CC_NP_VP_., IN...	70k/7k/7k
linguisticprobing/tree_depth	[depth_5, depth_6, depth_7, depth_8, depth_9, ...	85k/9k/9k
linguisticprobing/coordination_inversion	[I, O]	100k/10k/10k
linguisticprobing/odd_man_out	[C, O]	83k/8k/8k
linguisticprobing/bigram_shift	[I, O]	100k/10k/10k
snips_built_in_intents/default	[ComparePlaces, RequestRide, GetWeather, Searc...	328
amazon_polarity/amazon_polarity	[negative, positive]	4M/400k
winograd_wsc/wsc285	[0, 1]	285
winograd_wsc/wsc273	[0, 1]	273
hover/default	[NOT_SUPPORTED, SUPPORTED]	18k/4k/4k
dbpedia_14/dbpedia_14	[Company, EducationalInstitution, Artist, Athl...	560k/70k
onestop_english/default	[ele, int, adv]	567
movie_rationales/default	[NEG, POS]	2k/200/199
hans/plain_text	[entailment, non-entailment]	30k/30k
sem_eval_2014_task_1/default	[NEUTRAL, ENTAILMENT, CONTRADICTION]	5k/4k/500
eraser_multi_rc/default	[False, True]	24k/5k/3k
selqa/answer_selection_experiments	[0, 1]	66k/19k/9k
scitail/tsv_format	[entailment, neutral, contradiction]	23k/2k/1k

B Task Embedding Stability

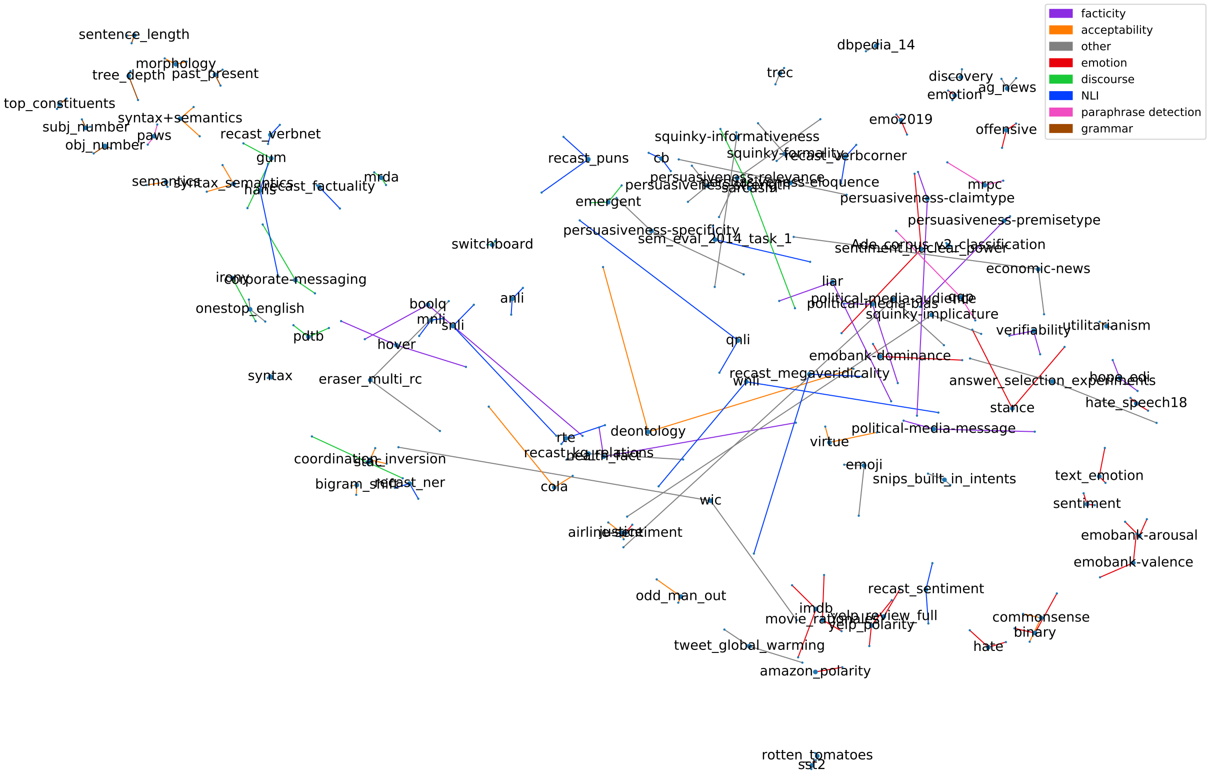

Figure 4: UMAP Visualization of task embeddings when each task is attributed 3 task embeddings. For each task, we position the task name at the centroid of the three embeddings and represent edges between the centroid and the two other embeddings.

C PCA Visualization of Task Embeddings

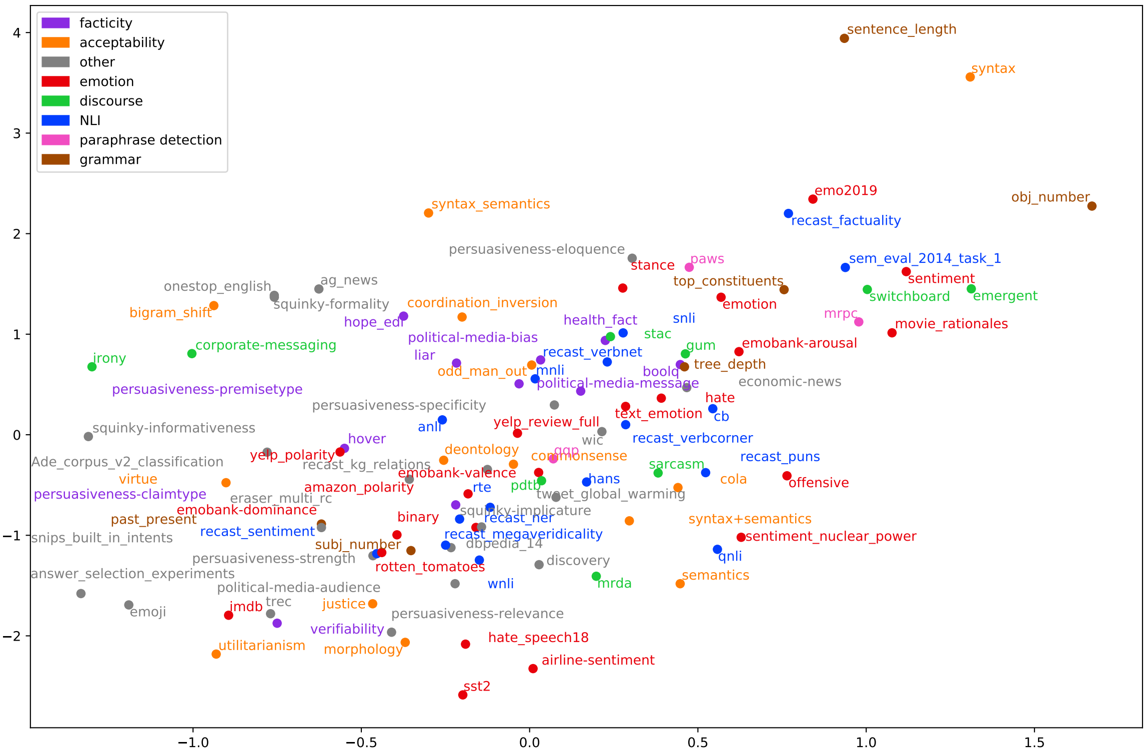

Figure 5: PCA Visualization of task embeddings.