Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), pages 633–647
Marseille, 20-25 June 2022

Analysis and Prediction of NLP models via Task Embeddings

Damien Sileo, Marie-Francine Moens
KU Leuven
Belgium
damien.sileo@kuleuven.be, sien.moens@cs.kuleuven.be

Abstract

Task embeddings are low-dimensional representations that are trained to capture task properties. In this paper, we propose MetaEval, a collection of 101 NLP tasks. We fit a single transformer to all MetaEval tasks jointly while conditioning it on learned embeddings. The resulting task embeddings enable a novel analysis of the space of tasks. We then show that task aspects can be mapped to 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.

Keywords: task embeddings, metalearning, natural language processing, evaluation, extreme multi-task learning

1. Introduction

Transfer between tasks enabled considerable progress in NLP. Pretrained transformer-based encoders, such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), 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; Warstadt et al., 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). These weights provide a way to predict relatedness between tasks (Achille et al., 2019; Vu et al., 2020). However, they are very high dimensional, and they cannot be modulated to adapt the network to unseen tasks. Hypernetworks (Ha et al., 2017; von Oswald et al., 2020; Hansen et al., 2020), i.e., neural networks whose weights are modulated by an outer network, can solve this problem. These techniques have been adapted to NLP by Pilault et al. (2021) and Mahabadi et al. (2021) who rely on adapters (Houlsby et al., 2019). Adapters are parameter-efficient layers that can be inserted between specific layers and trained to modulate a frozen transformer. An adapter A_i is composed of distinct adapter layers. Pilault et al. (2021) and Mahabadi et al. (2021) 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 $A(z_i)$. The task embeddings and conditional adapter are trained jointly, which allows each task to modulate the shared model in its own way. This approach leads to a performance improvement over individual adapters or full fine-tuning while allowing very low-dimensional (dim(z) < 100) task representations.

In this work, we leverage conditional adapters to derive task embeddings for 101 tasks based on a joint multitask training objective. This enables new analyses of the relationships among the tasks. We show that we can predict the task embeddings from selected task aspects, which leads to a more selective and interpretable control of NLP models.

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. 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].

2. Related Work

Task relatedness and task embeddings A common way to measure task relatedness is to train a model on

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

[2] `https://github.com/sileod/metaeval`

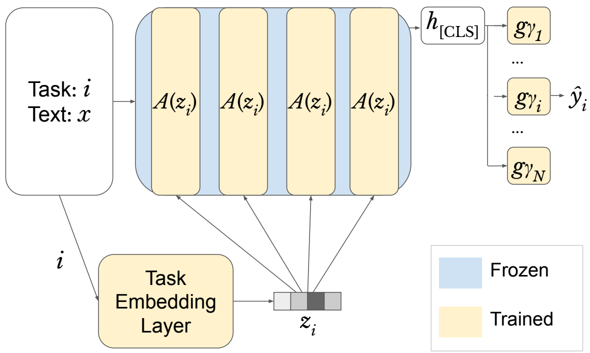

Figure 1: An overview of a transformer with a conditional adapter in a classification setup with N tasks. Batches for each task are used sequentially in random order. Each text example x is represented by $h_{[\text{CLS}]}$, which is the input of g_{γ_i} and the classifier for the task $\mathcal{T}_i$.

a source task, or a combination of source tasks in the case of multitask learning (Caruana, 1997), and then measure the effect on 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. 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 on subtasks. We perform a comparable 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 transferability of task on each other, but we evaluate the properties of tasks in the latent space. Task embeddings were formalized and linked to task relatedness in computer vision tasks by Achille et al. (2019), who interpret pooled Fisher information in convolutional neural networks as task embedding. They treat each label as a task and compare task embeddings with labels. Vu et al. (2020) adapt this task embeddings technique to NLP models but they limit their analysis that to the prediction of task relatedness. Here, we also evaluate Fisher embeddings in the NLP context but also compare them to conditional adapter embeddings and probe task properties.

Probing neural text representations 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.

Transfer techniques Several alternatives were proposed to overcome the shortcomings of full fine-tuning. Houlsby et al. (2019) proposed adapters as a compact transformation to modulate a model without fine-tuning the whole network. Stickland and Murray (2019) leverage adapters in a multi-task setting with a fine-tuned transformer and task-specific adapters. Pilault et al. (2021) and Mahabadi et al. (2021) modulate a single adapter with task embedding to enable efficient multi-task learning and compact task representation, but do not perform inference on new tasks. von Oswald et al. (2020) propose a task inference model based on input data for continual learning problems on vision tasks, and Hansen et al. (2020) also applies this idea reinforcement learning for visual tasks. Cao and Yogatama (2020) address language generation on a variety of domains, which can be treated as tasks. They also rely on input data to predict task embeddings. Here, we adapt the idea of task inference from input data to NLP classification tasks, but we also show that known task attributes such as task type can be used instead of the input data. This is analogous to Üstün et al. (2020) who use typological language features for adaptation of dependency parsing to new languages. Finally, prompts can be also used for transfer without fine-tuning (Radford et al., 2019) or by tuning token embeddings to learn a prompt (Li and Liang, 2021; Qin and Eisner, 2021), but they are used for text generation or knowledge probing which are outside the scope of this work.

3. Models and Setups

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] d-dimensional 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 $\text{H}(y_i, g_{\gamma_i}(f_\theta(x)))$ where y denotes a label. 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

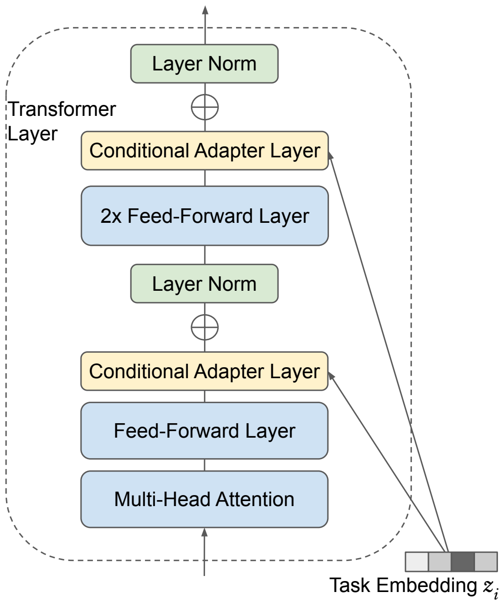

Figure 2: A transformer layer with conditional adapter layers.

only the adapter weights and classifier weights (α_i, γ_i) for each task.

Conditional Adapters We replace task-specific adapters with conditional adapters $A_\alpha(z_i)$ that are common to all tasks but conditioned on task embeddings z_i. To do so, we train all the tasks jointly and optimize 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:

$$\min_{(\alpha,z,\gamma)} \sum_{\mathcal{T}_i \in \Theta} \mathrm{H}(y_i, \hat{y}_i)$$

3.1. Parametrization of Adapters and Conditional Adapters

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

$$h \to h + \mathrm{LayerNorm}_{\gamma,\beta}(U(\mathrm{GeLU}(D(h)))) \quad (1)$$

where D and U are linear down-projection and up-projection matrices in $\mathbb{R}^{d\times a}$ and $\mathbb{R}^{a\times d}$ respectively. Adapter layers are inserted between fixed-weight transformer layers to adjust the text representation for the target task. Layer normalization weights γ, β (Ba et al., 2016) are also optimized and are considered as a part of the adapters.

In a conditional adapter, LayerNorm weights are modulated by z in the following way: $\gamma, \beta = W_\gamma z, W_\beta z$ where W_γ and W_β are learnable randomly initialized projections in $\mathbb{R}^{\dim(z)\times d}$.

Mahabadi et al. (2021) use a similar modulation of the D, U matrices and generate their weight: $D, U = W_D z, W_U z$ with $W_D \in \mathbb{R}^{(d\times a)\times \dim(z)}$, $W_U \in \mathbb{R}^{(a\times d)\times \dim(z)}$. They also show that adapters can be shared across layers, but this did not lead to improvement in our experiments.

Instead, Pilault et al. (2021) use the following transformation[3]:

$$h \to h + W_h z \odot h + W_b z \quad (2)$$

where W_h and W_b are projections in $\mathbb{R}^{\dim(z)\times d}$, before each adapter layer.

3.2. Baseline task embeddings

We also perform experiments with the task embeddings methods proposed by Vu et al. (2020) instead of a learned task embedding. We project them to $\dim(z)$ with a randomly initialized trainable linear layer.

TextEmb is the average text embedding across all examples of a task. We use the average of the output tokens (Vu et al., 2020; Reimers and Gurevych, 2019) as text embeddings.

Fisher Embedding captures the influence of the training objective on the activation of $h_{[\mathrm{CLS}]}$. See appendix A for additional details.

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[4] 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.

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.

[3] Pilault et al. (2021) also have proposed a conditional attention which did not yield improvement in our experiments

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

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 (Mahabadi et al., 2021)	75.6	38M	512
Conditional Adapter (Pilault et al., 2021)	**79.7**	10M	32
z=TextEmb task embedding (Vu et al., 2020)	69.9	10M	32
z=Fisher information task embedding (Vu et al., 2020)	67.5	10M	32

Table 1: Parameter counts and MetaEval test accuracy percentages of fine-tuning techniques. The last two rows replace the latent task embedding z with a linear projection of the task features proposed by Vu et al. (2020).

PragmEval (Sileo et al., 2019a) 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, present words, parts of speech, 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 B 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$_{\text{BASE}}$ (Liu et al., 2019) pretrained transformer[5] , a sequence length of 128, a batch size of 64, and Adam with a learning rate of 2.10^{-5} as an optimizer during 3 epochs for single-tasks model, 1 epoch while multitasking and an adapter size $a = 256$ with (Pilault et al., 2021) and $a = 32$ with (Mahabadi et al., 2021) as they suggest. We use the same hyperparameters for the baselines otherwise (tuning them did not lead to significant improvement). We set a limit of $30k$ training examples per task per epoch to obtain manageable computation time.

Multitask setup When multitasking, we sample one task from among all MetaEval tasks at each training step. The loss for each task is capped to 1.0 to prevent unbalance between tasks. We also sample each task with a probability proportional to the square root of the dataset size (Stickland and Murray, 2019) to balance the mutual influence of the tasks. We use task embeddings of $\dim(z) = 32$, which was selected according to MetaEval average validation accuracy among $\{2, 8, 32, 128, 512\}$ and is also suggested by Mahabadi et al. (2021).

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 adapters achieve 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. We use the model of Pilault et al. (2021) with latent task embeddings from now on because of its higher performance.

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[6] projection, which is less readable but provides a more faithful depiction of the global structure, is shown

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

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

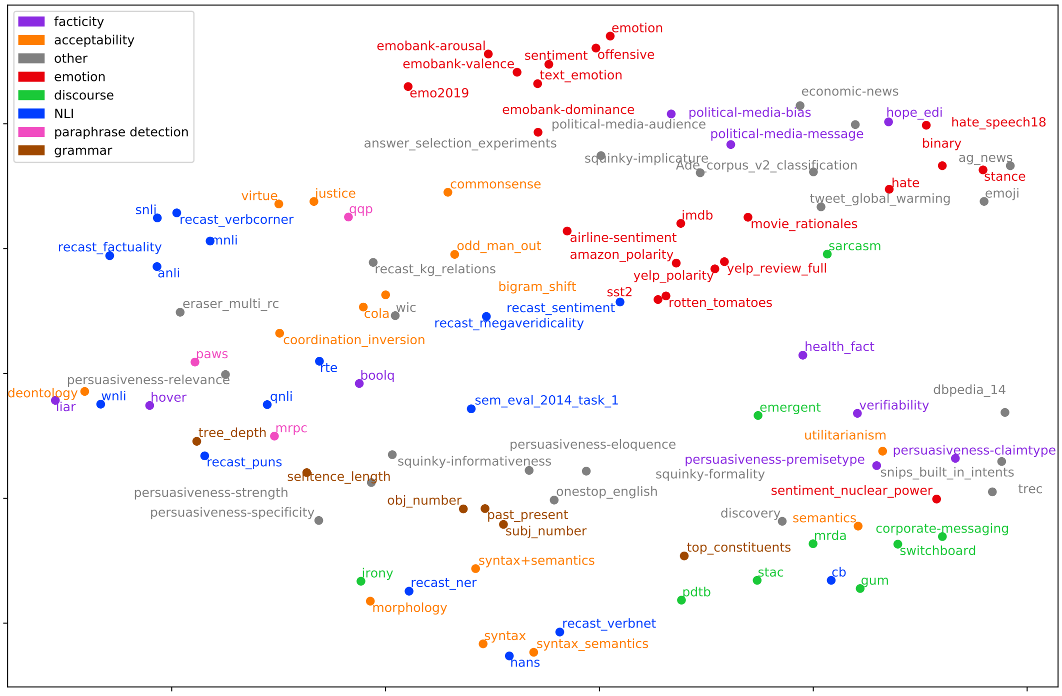

Figure 3: UMAP projection of the task embeddings.

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 type. As a reference, we provide the expected stability that would be obtained for randomly sampled task embedding positions.

in Appendix D. 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.

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 randomly selected for each task training step.
Figure 4 in Appendix C 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 portion of the embedding space. For each task, we compute the rate at which the 10 nearest neighbors [7] 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

[7] According to cosine similarity.

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.

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 Λ_i, we 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[8] 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.[9]

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, FACTICITY, GRAMMAR, NLI, PARAPHRASE DETECTION, OTHER }.

Note that the above aspects do not rely on annotated data (only on the input text, sizes, and task type). We use a logistic regression classifier with Scikit-Learn (Pedregosa et al., 2011) default parameters[10] 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. Fisher Embeddings perform poorly, but Vu et al. (2020) explain that, unlike other methods, the Fisher embeddings do not lie in a comparable space. TextEmb performs surprisingly well in these probing tasks, however, it does not fully capture the task type, since concatenation with the latent embedding improves the classification of this aspect. This can explain the relatively low performance of TextEmb in table 1. The task embeddings do not seem to accurately capture the difference between single sentence or sentence pair tasks, except for TextEmb, which is sensitive to separator tokens.

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 with the MetaEval multitask setup and then train a regression model to predict the task embeddings from the task aspects $\Lambda_{\mathcal{T}_i}$ or TextEmb.

$$\hat{z}_i = \text{Regression}([a, a \in \Lambda_{\mathcal{T}_i}]) \qquad (3)$$

To evaluate task embedding regression, 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:

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

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

[10]Release 0.24.1; deviation from the default parameters did not lead to a significant improvement. We also experimented with gradient boosting trees and KNN classifiers with no improvement.

Features	Domain-Cluster	Num-Rows	Num-Text-Fields	Task-Type	Text-Length	AVG
n.a. (Majority Class)	27.8	**62.4**	63.4	20.8	24.8	39.8
Fisher Embedding	37.7	61.3	62.2	35.7	25.6	44.5
Latent Features	41.8	51.5	62.2	45.7	32.8	46.8
Latent Features$\oplus$TextEmb	71.5	60.4	76.3	**60.4**	53.6	64.4
TextEmb	**78.3**	59.3	**82.3**	50.6	**60.3**	**66.2**

Table 4: Accuracy of task aspect classification from task embeddings. $\oplus$ denotes concatenation.

	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	73.5	**93.6**	68.8	55.3	**72.7**	51.5	**70.2**	69.4
Same Task-Type Task Embeddings	**76.7**	91.4	67.6	57.0	67.0	53.8	64.0	68.2
Offline Task Embedding Ridge Regression	76.2	92.0	67.6	61.6	71.7	53.8	68.6	70.2
Features-Aware Task Embeddings - TextEmb (Vu et al., 2020)	75.4	90.2	70.2	53.9	66.5	57.3	65.8	68.1
Features-Aware Task Embeddings - Aspects (ours)	75.4	90.0	**70.4**	**71.1**	66.2	**56.2**	63.7	**70.4**

Table 5: 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 each 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.

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.

Features-Aware Task Embeddings We propose another variation, where we perform multitask training and the regression of embeddings jointly. Instead of using only a latent task embedding z_i for each task $\mathcal{T}_i$, we add it to a projection of the input features ϕ_i, which can be either a concatenation of all aspects[11], or TextEmb. The task embedding modulating the adapters is then $z_i + W_\phi \phi_i$. An unseen task $\mathcal{T}_i$ can be represented by the projection from aspect embeddings augmented with the average latent task embedding.

As another 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 5 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 is achieved via offline regression. Using TextEmb as features performs worse than using latent embeddings which indicates that TextEmb does not capture enough important information about the task type[12]. 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. These findings address **RQ3** and establish task embeddings as a viable gateway for zero-shot transfer of NLP tasks based on task attributes.

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 on task aspects. 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, such as undesirable associations language.

7. Acknowledgements

This work is part of the CALCULUS project, which is funded by the ERC Advanced Grant H2020-ERC-2017. ADG 788506[13].

[11]Using one-hot representations.

[12]Fisher task embeddings led to lower accuracies.

[13]`https://calculus-project.eu/`

8. Bibliographical References

Achille, A., Lam, M., Tewari, R., Ravichandran, A., Maji, S., Fowlkes, C. C., Soatto, S., and Perona, P. (2019). Task2vec: Task embedding for meta-learning. In *Proceedings of ICCV2019*, pages 6430–6439.

Aghajanyan, A., Zettlemoyer, L., and Gupta, S. (2020). Intrinsic Dimensionality Explains the Effectiveness of Language Model Fine-Tuning.

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

Ba, L. J., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization. *CoRR*, abs/1607.06450.

Barbieri, F., Camacho-Collados, J., Espinosa-Anke, L., and Neves, L. (2020). TweetEval:Unified Benchmark and Comparative Evaluation for Tweet Classification. In *Proceedings of Findings of EMNLP*.

Cao, K. and Yogatama, D. (2020). Modelling latent skills for multitask language generation. *CoRR*, abs/2002.09543.

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

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., and Bordes, A. (2017). Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of EMNLP2017*, pages 670–680, Copenhagen, Denmark, September. Association for Computational Linguistics.

Conneau, A., Kruszewski, G., Lample, G., Barrault, L., and Baroni, M. (2018). What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia, 7. Association for Computational Linguistics.

D'Amour, A., Heller, K., Moldovan, D., Adlam, B., Alipanahi, B., Beutel, A., Chen, C., Deaton, J., Eisenstein, J., Hoffman, M. D., and others. (2020). Underspecification presents challenges for credibility in modern machine learning. *arXiv preprint arXiv:2011.03395*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL2019*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Ha, D., Dai, A. M., and Le, Q. V. (2017). Hypernetworks. In *Proceedings of ICLR2017*. OpenReview.net.

Hansen, S., Dabney, W., Barreto, A., Warde-Farley, D., de Wiele, T. V., and Mnih, V. (2020). Fast task inference with variational intrinsic successor features. In *Proceedings of ICLR2020*.

Hendrycks, D. and Gimpel, K. (2016). Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *CoRR*, abs/1606.08415.

Hendrycks, D., Burns, C., Basart, S., Critch, A., Li, J., Song, D., and Steinhardt, J. (2021). Aligning ai with shared human values. In *International Conference on Learning Representations*.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. (2019). Parameter-efficient transfer learning for nlp. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Li, X. L. and Liang, P. (2021). Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of ACL 2021*, pages 4582–4597, Online, August. Association for Computational Linguistics.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach.

Mahabadi, R. K., Ruder, S., Dehghani, M., and Henderson, J. (2021). Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *Proceedings of ACL2021*.

McInnes, L., Healy, J., and Melville, J. (2018). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *arXiv preprint arXiv:1802.03426*, February.

Meng, Y., Huang, J., Wang, G., Zhang, C., Zhuang, H., Kaplan, L., and Han, J. (2019). Spherical text embedding. In *Proceedings of NIPS2019*.

Mou, L., Meng, Z., Yan, R., Li, G., Xu, Y., Zhang, L., and Jin, Z. (2016). How transferable are neural networks in NLP applications? In *Proceedings of EMNLP2016*, pages 479–489, Austin, Texas, November. Association for Computational Linguistics.

Nayak, N., Angeli, G., and Manning, C. D. (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, August. Association for Computational Linguistics.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Proceedings of JMLR2011*, 12:2825–2830.

Phang, J., Févry, T., and Bowman, S. R. (2018). Sentence Encoders on STILTs: Supplementary Training on Intermediate Labeled-data Tasks. *arXiv preprint arXiv:1811.01088v2*.

Pierrejean, B. and Tanguy, L. (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, May. Association for Computational Linguistics.

Pilault, J., Elhattami, A., and Pal, C. (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.

Poliak, A., Haldar, A., Rudinger, R., Hu, J. E., Pavlick, E., White, A. S., and Van Durme, B. (2018). Collecting diverse natural language inference problems for sentence representation evaluation. In *Proceedings EMNLP2018*, pages 67–81, Brussels, Belgium, October-November. Association for Computational Linguistics.

Qin, G. and Eisner, J. (2021). Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proceedings of the ACL 2021*, pages 5203–5212, Online, June. Proceedings of ACL2021.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of EMNLP2019*. Association for Computational Linguistics, 11.

Sia, S., Dalmia, A., and Mielke, S. J. (2020). Tired of topic models? clusters of pretrained word embeddings make for fast and good topics too! In *Proceedings of EMNLP2020*, pages 1728–1736, Online, November. Association for Computational Linguistics.

Sileo, D., de Cruys, T. V., Pradel, C., and Muller, P. (2019a). Discourse-based evaluation of language understanding. *arXiv preprint arXiv:2007.09604*.

Sileo, D., Van De Cruys, T., Pradel, C., and Muller, P. (2019b). Mining discourse markers for unsupervised sentence representation learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3477–3486, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Stickland, A. C. and Murray, I. (2019). BERT and PALs: Projected attention layers for efficient adaptation in multi-task learning. In Kamalika Chaudhuri et al., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5986–5995, Long Beach, California, USA, 09–15 Jun. PMLR.

Tenney, I., Das, D., and Pavlick, E. (2019). BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of ACL2019*, pages 4593–4601, Florence, Italy, 7. Association for Computational Linguistics.

Üstün, A., Bisazza, A., Bouma, G., and van Noord, G. (2020). UDapter: Language adaptation for truly Universal Dependency parsing. In *Proceedings of EMNLP2020*, pages 2302–2315, Online, November. Association for Computational Linguistics.

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

von Oswald, J., Henning, C., Sacramento, J., and Grewe, B. F. (2020). Continual learning with hypernetworks. In *Proceedings of ICLR2020*.

Vu, T., Wang, T., Munkhdalai, T., Sordoni, A., Trischler, A., Mattarella-Micke, A., Maji, S., and Iyyer, M. (2020). Exploring and predicting transferability across NLP tasks. In *Proceedings of EMNLP2020*, pages 7882–7926, Online, November. Association for Computational Linguistics.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. (2019a). Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in neural information processing systems*, pages 3266–3280.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. (2019b). GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *International Conference on Learning Representations*.

Warstadt, A., Cao, Y., Grosu, I., Peng, W., Blix, H., Nie, Y., Alsop, A., Bordia, S., Liu, H., Parrish, A., Wang, S., Phang, J., Mohananey, A., Htut, P. M., Jeretic, P., and Bowman, S. R. (2019). Investigating BERT's Knowledge of Language: Five Analysis Methods with NPIs. In *EMNLP/IJCNLP*.

Warstadt, A., Parrish, A., Liu, H., Mohananey, A., Peng, W., Wang, S.-F., and Bowman, S. R. (2020a). BLiMP: The benchmark of linguistic minimal pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Warstadt, A., Zhang, Y., Li, X., Liu, H., and Bowman, S. R. (2020b). Learning Which Features Matter: RoBERTa Acquires a Preference for Linguistic Generalizations (Eventually). In *Proceedings of EMNLP2020*, pages 217–235, Online, 11. Association for Computational Linguistics.

Wendlandt, L., Kummerfeld, J. K., and Mihalcea, R. (2018). Factors influencing the surprising instability of word embeddings. In *Proceedings of NAACL2018*, pages 2092–2102, New Orleans, Louisiana, June. Association for Computational Linguistics.

Wolf, T., Lhoest, Q., von Platen, P., Jernite, Y., Drame, M., Plu, J., Chaumond, J., Delangue, C., Ma, C., Thakur, A., Patil, S., Davison, J., Scao, T. L., Sanh, V., Xu, C., Patry, N., McMillan-Major, A., Brandeis, S., Gugger, S., Lagunas, F., Debut, L., Funtowicz, M., Moi, A., Rush, S., Schmidd, P., Cistac, P., Muštar, V., Boudier, J., and Tordjmann, A. (2020). Datasets. *GitHub. Note: https://github.com/huggingface/datasets*, 1.

Yang, D., Lavie, A., Dyer, C., and Hovy, E. (2015). Humor recognition and humor anchor extraction. In *Proceedings of EMNLP2015*, pages 2367–2376, Lisbon, Portugal, September. Association for Computational Linguistics.

A. Fisher Information Task Embedding

The average of text embeddings for all samples of a task can be used as a task embedding, but they ignore the labels entirely. Achille et al. (2019) propose a task embedding based on the influence of a task training objective on network weights. To do so, they use an empirical Fisher information estimate of a fine-tuned network as a task embedding. Fisher information captures the influence of model parameters or activations on the loss function. For BERT-based models, Vu et al. (2020) suggest the use of $h_{[\text{CLS}]}$ token activation and to only consider the diagonal information of the Fisher matrix, which is the expected variance of the gradients of the log-likelihood with respect to activations. Activation dimensions that are important for a task will have a high fisher information. Since similar tasks should use similar features, Fisher information of the activations can capture useful task representations. As suggested by Vu et al. (2020), we perform a fine-tuning for each task before task embedding computation with the setting of section 5.2. We then compute the empirical Fisher information embeddings for a task i as follows:

$$F_\theta(\mathcal{T}_i) = \frac{1}{n}\sum_{k=1}^{n}(\nabla_\theta \log P_\theta(y_k, x_k))^2 \tag{4}$$

Where n is the number of training samples. θ can be the full network or any activation but here we use the $h_{[\text{CLS}]}$ activation, which achieved the best results in our section 5.2 experiment.

B. 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

Continued on next page

Dataset	Labels	Splits Sizes
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
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

Continued on next page

Dataset	Labels	Splits Sizes
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

C. Task Embedding Stability

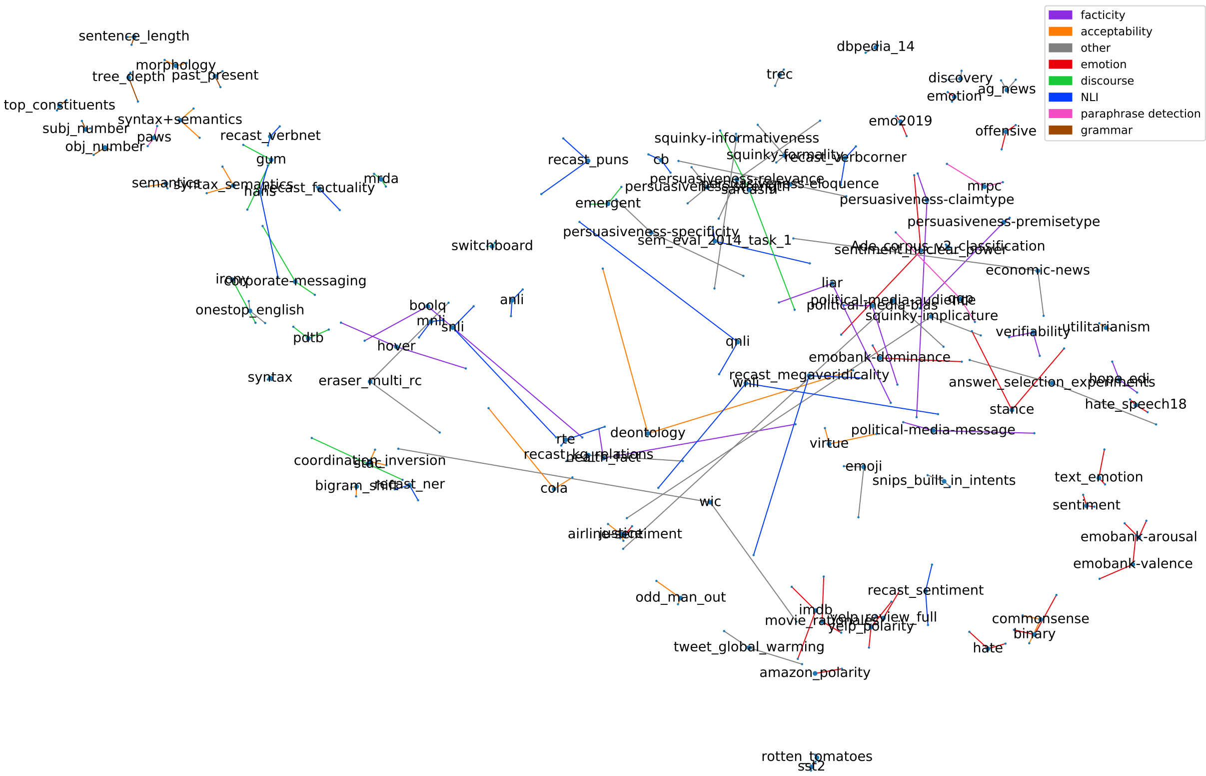

Figure 4: UMAP Visualization of task embeddings when each task is attributed 3 task embeddings. For each task, we position the task name at the centroid of the three embeddings and represent edges between the centroid and the two other embeddings.

D. PCA Visualization of Task Embeddings

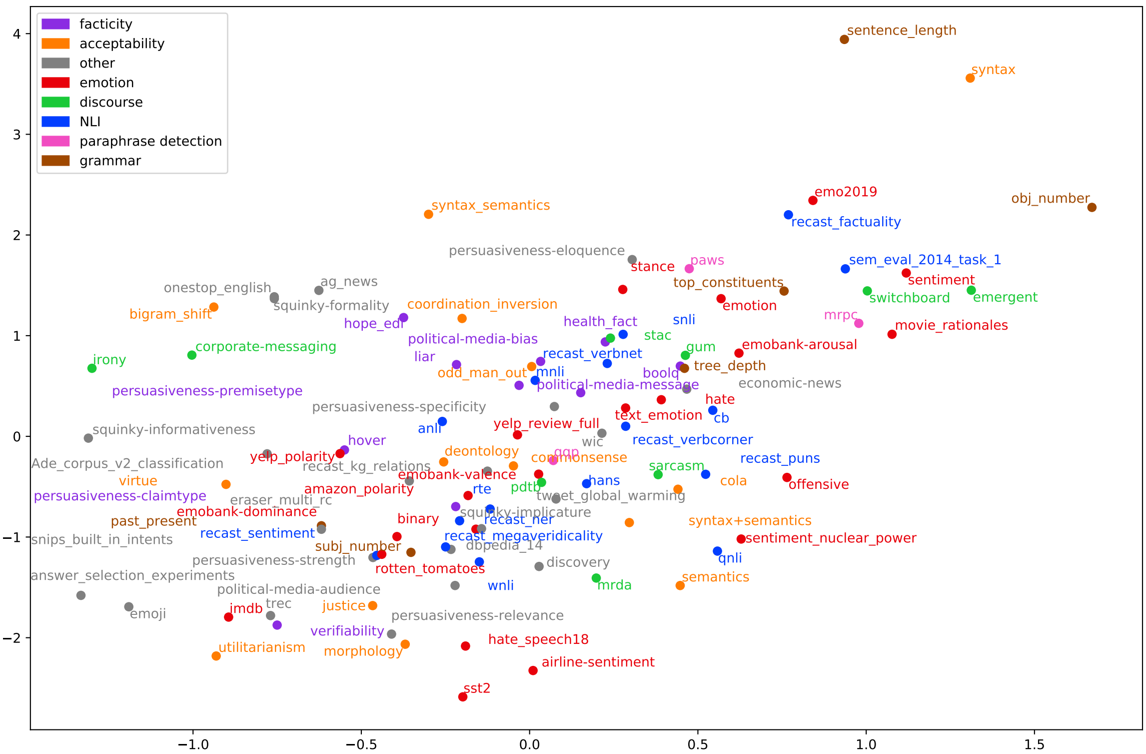

Figure 5: PCA Visualization of task embeddings.