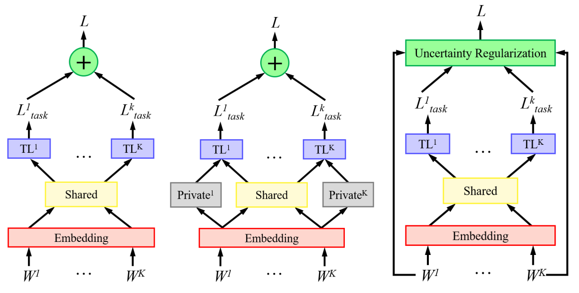
Uncertainty Regularized Multi-Task Learning

Kourosh Meshgi and **Maryam Sadat Mirzaei** and **Satoshi Sekine**
RIKEN Center for Advanced Intelligence Project (AIP)
Tokyo, Japan
{kourosh.meshgi, maryam.mirzaei, satoshi.sekine}@riken.jp

Abstract

By sharing parameters and providing task-independent shared features, multi-task deep neural networks are considered one of the most interesting ways for parallel learning from different tasks and domains. However, fine-tuning on one task may compromise the performance of other tasks or restrict the generalization of the shared learned features. To address this issue, we propose to use task uncertainty to gauge the effect of the shared feature changes on other tasks and prevent the model from overfitting or over-generalizing. We conducted an experiment on 16 text classification tasks, and findings showed that the proposed method consistently improves the performance of the baseline, facilitates the knowledge transfer of learned features to unseen data, and provides explicit control over the generalization of the shared model.



(a) FS-MTL (b) SP-MTL (c) UFS-MTL

Figure 1: Different architectures for multi-task learning (MTL) for text classification with the LSTM baseline. (a) Fully-Shared MTL in which the shared layer provides a shared feature space and task-layers (TLs) convert them into final task outputs, (b) Shared-Private MTL, where tasks jointly learn a shared feature set while having their own (private) features, (c) Uncertainty-regularized FS-MTL (proposed) in which the uncertainty of all tasks are measured while fine-tuning for each task to grant more generalization to the learned shared features.

1 Introduction

Multi-task learning (MTL) is a branch of supervised learning that strives to improve the generalization of the regression or classification task by leveraging the domain-specific information contained in the training signals of related tasks (Caruana, 1993). MTL has been investigated in various applications of machine learning, from natural language processing (Collobert and Weston, 2008; Clark et al., 2019) and speech recognition (Deng et al., 2013; Suthokumar et al., 2020) to computer vision (Girshick, 2015; Zamir et al., 2018). The tasks can be defined as applying the same model on different data (also known as multi-domain learning) (Nam and Han, 2016; Liu et al., 2017a), or on various problems (e.g., named entity recognition, entity mention detection and relation extraction in HMTL (Sanh et al., 2019)).

When training a multi-task learner, training each task normally increases its accuracy (fine-tuning) and, at the same time, provides more information for the shared representation that affects the accuracy of the rest of the tasks (generalization). Balancing the finetuning-generalization trade-off has been the subject of several studies. Kendall et al. (2018) adjusts tasks' relative weights in the loss function in proportion to the task uncertainty. Liu et al. (2016) divides the feature space into task-specific and shared spaces and later employs adversarial learning to encourage shared feature space to contain more common information and no task-specific information (Liu et al., 2017a). Bousmalis et al. (2016) proposed orthogonality constraints to punish redundancy between shared and task-specific layers. In line with this direction, learning through hints (Abu-Mostafa, 1990) directly trains a network to predict the most important features. Yet, none of those methods explicitly balances the fine-tuning of the under-training task with its effect on the other tasks.

Here, we propose a method that considers the generalization of all tasks along with the task-specific loss function. As a good indicator of how other tasks are affected by the change of the shared feature space, we proposed measuring the learner's uncertainty on each task. Task uncertainty captures the relative confidence between tasks and reflects the inherent uncertainty in each task (Kendall et al., 2018). Therefore, the objective of the proposed MTL would be to maximally fine-tune each task on its corresponding training data while keeping the uncertainty of other tasks to the minimum. In other words, the MTL is expected to maintain a low level of the overall uncertainty and disentangle the training on the shared layers and task-specific layers. Our main contributions are

- Exploring task uncertainty to elicit more generalizable features in the shared layers,
- Conducting extensive ablation experiments to investigate the effects of different uncertainty metrics, pre-training, fine-tuning, using auxiliary tasks, and semi-supervised learning on the performance of this method,
- Experimenting on multi-domain and multi-task learning problems with homogeneous and heterogeneous tasks.

2 Related Works

Multi-Task Learning: MTL exchanges information learned by different tasks to improve overall performance. Such information may be obtained by jointly working with adversarial tasks (Ganin and Lempitsky, 2015), tasks working on different subsets of a common data pool (Meyerson and Miikkulainen, 2018), or tasks in a hierarchy (Sanh et al., 2019). Additionally, some tasks are serving as facilitators for harder or more complicated ones in various ways, such as providing hints/attention map (Yu and Jiang, 2016; Caruana, 1997), learning base representations (Rei, 2017; Subramanian et al., 2018) and preventing quick-plateaus during training (Bingel and Søgaard, 2017). Discovering the relationship between tasks or dynamically grouping them are other ways that MTL promotes the information transfer between tasks (Ruder et al., 2017; Zamir et al., 2018; Standley et al., 2019).

When used with deep learning, MTL models tend to share learned parameters across different tasks through *(i)* hard parameter sharing (Caruana, 1993; Kokkinos, 2017) in which the hidden layers are shared between all tasks, while several task-specific output layers are fine-tuned for each task, *(ii)* soft parameter sharing, in which each task has its model, and the distance between the parameters of the models for different tasks are regularized to encourage the parameters to be similar using, e.g., ℓ_1 norm (Duong et al., 2015) or trace norm (Yang and Hospedales, 2017), or *(iii)* partial parameter sharing, to avoid task interference and leverage task commonalities among a subset of the tasks (Zaremoodi et al., 2018; Rosenbaum et al., 2018).

In the hard parameter sharing architectures, shared parameters provide a global feature representation, while task-specific layers further process these features or provide a complementary set of features suitable for a specific task. Some MTL approaches are based on the intuition that learning easy tasks is the prerequisite for learning more complex ones (Ruder, 2017), hence put tasks in hierarchies (Søgaard and Goldberg, 2016; Hashimoto et al., 2017; Sanh et al., 2019) or try to automatically group similar tasks to dynamically form shared layers (Liu et al., 2017b).

Task Uncertainty: In an MTL setting, multiple tasks are intermittently trained and modify the shared parameters to minimize their loss (the losses can be back-propagated at once as well). This change affects how other tasks behave in various ways, one of which is the amount of uncertainty that each task bears. Uncertainty signals the information that the model lacks, or the sort of information that cannot be inferred from data (Kendall and Gal, 2017). There are various ways to measure uncertainty. Kendall et al. (2015) measures uncertainty via drop-out sampling. Later, Kendall et al. (2018) proposed Homoscedastic uncertainty to measure the uncertainty of entire tasks independent of the data. In another attempt, Kampffmeyer et al. (2016) computes the standard deviation of softmax outputs and average them to quantify the uncertainty of all tasks in MTL. Other approaches that leverage uncertainty to reduce the overfitting in MTL framework are presented in (Uma et al., 2020; Fornaciari et al., 2021).

3 Multi-task Classification

Neural text classification has been studied as one of the fundamental NLP problems. Some researchers replace hand-crafted features with with word-level and character-level representations obtained by CNNs (Kim, 2014; Zhang et al., 2015), others use RNNs, Convolutional RNNs and Self-attentive LSTMs for sequence modeling (Liu et al., 2016;

Lai et al., 2015; Liu and Guo, 2019). To highlight the effect of the proposed uncertainty regularization, here we use a simple LSTM-based text classifier as in (Jozefowicz et al., 2015).

3.1 Baseline Classifier

The text sequence $\mathbf{w} = \{w_1, w_2, \ldots, w_T\}$ is converted to a sequence of word embeddings $\mathbf{x}_i$ and is given to an LSTM layer. Each unit of LSTM layer at time t, includes an input gate $\mathbf{i}_t$, a forget gate $\mathbf{f}_t$, an output gate $\mathbf{o}_t$, a memory cell $\mathbf{c}_t$ and a hidden state $\mathbf{h}_t$. The LSTM implements

$$\begin{bmatrix} \tilde{\mathbf{c}}_t \\ \mathbf{o}_t \\ \mathbf{i}_t \\ \mathbf{f}_t \end{bmatrix} = \begin{bmatrix} \tanh \\ \sigma \\ \sigma \\ \sigma \end{bmatrix} \left(\mathbf{W}_p \begin{bmatrix} \mathbf{x}_t \\ \mathbf{h}_{t-1} \end{bmatrix} + \mathbf{b}_p \right) \tag{1}$$

$$\mathbf{c}_t = \tilde{\mathbf{c}}_t \odot \mathbf{i}_t + \mathbf{c}_{t-1} \odot \mathbf{f}_t$$
$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t)$$

in which $\sigma(.)$ and $\tanh(.)$ are logistic sigmoid and hyperbolic tangent functions, $\mathbf{W}_p$ and $\mathbf{b}_p$ are the weights and biases of LSTM (summarized in θ_p), and $\odot$ represents element-wise multiplication. The LSTM is then updated as

$$\mathbf{h}_t = \mathrm{LSTM}(\mathbf{h}_{t-1}, \mathbf{x}_t, \theta_p) \tag{2}$$

where the output of the last unit $\mathbf{h}_T$ represents the whole sequence. This is then fed to the task-specific output layers. The network is then trained on a training corpus with N samples $(\mathbf{w}_i, y_i)$ using cross-entropy loss function

$$L(\hat{y}, y) = -\sum_{i=1}^{N} \sum_{j=1}^{C} y_i^j \log\left(\hat{y}_i^j\right) \tag{3}$$

where y_i^j is the groundtruth inside $\{1..C\}$ and $\hat{y}_i^j$ is the predicted probability of label j for document i.

3.2 Multi-Task Learning Formalization

MTL aims to promote learning efficiency and overall task performances by exploiting commonalities and shared structures among tasks. In a neural net MTL, each task shares a portion of its parameters with a few or all other tasks to benefit from extra training they might get from those parallel tasks.

In an MTL text classification, the tasks may share all parameters of certain layers. In this fully-shared setting (Figure 1(a)), the shared LSTM layers are shared between all tasks to extract similar features. MTL training in this setting optimizes these features such that they are useful for all tasks. We used this approach in this study. Another way of parameter sharing is to share a common feature extractor (shared LSTM), but on top of that, each task has its own private feature extractor to complement the shared features (Figure 1(b)) as proposed in (Liu et al., 2017a).

4 Proposed Method

Let's assume an MTL with K tasks, in which each task k has a dataset D_k with N_k samples, where $D_k = \left\{\left(w_i^k, y_i^k\right)\right\}_{i=1}^{N_k}$. To obtain a probability distribution $\hat{y}^{(k)}$ for labels of task k, the shared feature $\mathbf{h}_T^{(k)}$ are fed to the final task-specific softmax layer. We train the network by minimizing the cross entropy loss between predicted and true label distributions ($\hat{y}^{(k)}$ and $y^{(k)}$) as follows:

$$L_{task} = \sum_{k=1}^{K} \alpha_k L\left(\hat{y}^{\langle k \rangle}, y^{\langle k \rangle}\right) \tag{4}$$

where α_k is the task importance coefficient and $L(\hat{y}, y)$ is defined in eq(3).

4.1 Proposed Uncertainty Regularization

Learning model uncertainty can be attributed to the uncertainty of the model due to the lack of training data and the information that the data cannot explain. The latter can be either *(i)* data-dependent that is reflected by observing the model output and *(ii)* data-independent that varies between different tasks (Kendall et al., 2018). MTL improves the learning over single-task learning by drawing on commonalities between inputs for different tasks to learn a shared representation, averaging on different task noises (Ruder, 2017), and exploiting the relations between tasks.

In a fully-shared MTL setting, each task contributes to the loss function based on errors it made, and since one task is being trained at a time, minimizing this error may negatively change the shared parameters for other tasks. This is usually alleviated by separating task-specific features from shared features using adversarial training and orthogonality constraints, yet the effect of the change in the shared layers on the performance of other tasks, while they are not being trained, is ignored.

A good MTL training procedure should be able to punish the changes in the shared feature space that increases the uncertainty of the task classifiers while only a single task is being fine-tuned to have better accuracy. Using uncertainty instead of task accuracy provides an additional signal to train the model. This helps by reflecting the internal state of the classifiers rather than their performance on a specific type of data, as the uncertainty signal includes both data- and task-dependent components. Using uncertainty regularization promotes the emergence of features that are more decisive to label the samples, and increases the overall performance of the classification. Coupled with overall task accuracy, the summation of uncertainty and task accuracy brings up features that are more independent and decisive, leading to the improvement of the MTL performance.

To calculate the uncertainty of a multi-class classifier, uncertainty sampling methods could be used. Thus, we proposed the uncertainty loss term as

$$L_{unc} = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{N_k}\sum_{i=1}^{N_k}\zeta(x_i), \tag{5}$$

where the uncertainty of all label predictions are averaged for each task, and $\zeta(.)$ can be calculated using least confidence (Settles and Craven, 2008)

$$\zeta_{LC} = 1 - P_k\left(\hat{y}^{(1)}|x\right), \tag{6}$$

or margin (Scheffer et al., 2001)

$$\zeta_M = 1 - P_k\left(\hat{y}^{(1)}|x\right) + P_k\left(\hat{y}^{(2)}|x\right), \tag{7}$$

or Shanon's entropy

$$\zeta_H = -2\sum_j P_k\left(\hat{y}^{(j)}|x\right) logP_k\left(\hat{y}^{(j)}|x\right); \tag{8}$$

where $\hat{y}^{(j)}$ is the label with jth largest predicted probability. The final loss function of the model can be written as

$$L = L_{task} + \lambda L_{unc} \tag{9}$$

in which λ is the regularization parameter.

4.2 Semi-Supervised Learning

As seen in eq(5), calculating the proposed uncertainty term L_{unc} is label-agnostic. This allows for using unlabeled data along with labeled ones during training. Therefore, the training data for each task can be augmented with synthetic data as well as data from other datasets (with matching statistics, context, and distribution).

4.3 Implementation Details

In our implementation we used 300D GloVe word embedding (Pennington et al., 2014) and a 128D LSTM with temporal dropout (0.5). The network weights are initialized with Xavier initialization, and the learning rate and regularization coefficients are selected by a grid search in the range $[0.001, 0.1]$ on the dev set (initial LR=0.01, λ =0.025). The tasks are trained in a round-robin order with mini-batches of size 16. For the rest of the procedure, we followed Søgaard and Goldberg (2016). We report the average of three independent runs of our method in experiment sections, and we used margin uncertainty (ζ_M) unless stated otherwise in all of our experiments.

5 Experiment

To evaluate our system, we considered two different settings to investigate the performance of our proposed system in multi-domain and multi-task settings. In the former case, a similar task is done on different datasets, while on the latter, several heterogeneous but related tasks are performed. The first experiment comes with extensive analysis on performance, generalization, uncertainty regularization, measuring uncertainty, convergence speed, internal dynamics, and system errors, as well as leveraging auxiliary tasks, unlabeled data, pre-training, and task fine-tuning.

5.1 Multiple Domains

In this experiment, we consider a homogeneous multi-task learning scenario in which 16 text classification tasks on various datasets are considered. Each dataset contains several reviews on the different products and movies with binary labels. After joint training on all domains (obtaining vanilla version which is the LSTM regularized by uncertainty loss), to conduct a fair comparison, we incorporated the additional modules that are also used in competitive models.Thus, we included fine-tuning, pre-training, and training on unlabeled data to obtain the final UFS-MTL. It has the state-of-the-art performance among MTL methods with the same LSTM baseline (Table 2), while the vanilla version itself leads to significant improvement in the performance as compared to LSTM which indicates the effectiveness of uncertainty regularization. Here, we used LSTM as a simple model to demonstrate the effectiveness of uncertainty, while other architectures such as RNN, CNN, and transformers can also use the benefits of this regularization. However, for the sake of simplicity we used LSTM to convey that using uncertainty improves the overall performance of the architecture.

Dataset: We took 14 product review datasets for different products, each serving as an individual domain from (Blitzer et al., 2007), and converted the labels to positive ($> 3\star$) or negative ($< 3\star$). We also take two movie review datasets, IMDB and

Datasets	Train	Dev	Test	Un.	Avg.Len	Vocab
Books	1400	200	400	2000	159	62K
Electronics	1398	200	400	2000	101	30K
DVD	1400	200	400	2000	173	69K
Kitchen	1400	200	400	2000	89	28K
Apparel	1400	200	400	2000	57	21K
Camera	1397	200	400	2000	130	26K
Health	1400	200	400	2000	81	26K
Music	1400	200	400	2000	136	60K
Toys	1400	200	400	2000	90	28K
Video	1400	200	400	2000	156	57K
Baby	1300	200	400	2000	104	26K
Magazines	1370	200	400	2000	117	30K
Software	1315	200	400	475	129	26K
Sports	1400	200	400	2000	94	30K
IMDB	1400	200	400	2000	269	44K
MR	1400	200	400	2000	21	12K

Table 1: Statistics of 16 datasets for multi-domain text classification experiment.

MR, with binary labels from (Maas et al., 2011), and (Pang and Lee, 2005) respectively. Each domain has approximately 2000 labeled comments with 70-10-20 split for train-dev-test dataset and 2000 unlabeled data (Table 1).

Competitor Models: We compared our algorithm with the vanilla LSTM baseline, MT-DNN (Liu et al., 2015) with bag-of-word representation and multi-layer perceptrons in which a hidden fully-connected layer is shared. We also compared it with MT-CNN (Collobert and Weston, 2008) with partially shared convolutional layers for different tasks, FS-MTL with word embedding and shared LSTM layers, as well as SP-MTL (Liu et al., 2016) in which a shared LSTM provides a part of feature representation for all tasks while each task has its private LSTM. Other comparisons include SSP-MTL (Chen et al., 2018) that stacks layers of SP-MTL, ASP-MTL (Liu et al., 2017a) that uses adversarial learning and orthogonality constraints to prevent the cross-interference of shared and private latent feature spaces in SP-MTL, and Meta-MTL (Chen et al., 2018) that uses a shared meta-network to capture the meta knowledge of semantic composition and generates the parameters of task-specific semantic composition models in SP-MTL.

Task-Specific Output Layer: The obtained shared representation is fed to the task-specific output classifiers composed of a fully connected layer followed by a softmax layer to predict the label

$$\hat{\mathbf{y}}^{\langle k \rangle} = \mathrm{softmax}\left(\mathbf{W}^{\langle \mathbf{k} \rangle}\mathbf{h}_T + \mathbf{b}^{\langle k \rangle}\right) \quad (10)$$

where $\mathbf{W}^{\langle k \rangle}$ and $\mathbf{b}^{\langle k \rangle}$ are the weights and biases of the task layer k and $\hat{\mathbf{y}}^{\langle k \rangle}$ is prediction probabilities.

Task Fine-Tuning: The training procedure selects mini-batches of all tasks intermittently. We can further optimize each task by freezing the shared layer and fine-tune each task individually. The results of this fine-tuning procedure are denoted by "+Fine" in Table 2.

Pre-Training: Initializing the shared layers with an unsupervised pre-training phase is a common practice. Thus, we initialize it by a language model (Bengio et al., 2007) which we trained on all of our dataset. Table 2 shows improvement in "+Pre".

Adding Auxiliary Task: One of the main challenges of sequence modeling is to capture semantic composition functions. Composition models can be sequential (Sutskever et al., 2014; Chung et al., 2014), convolutional (Collobert et al., 2011; Kalchbrenner et al., 2014), syntactic (Socher et al., 2013; Tai et al., 2015), and functional (Chen et al., 2018; Singh et al., 2021). Different compositional functions are learned from scratch in different tasks, while some tasks are more suitable in capturing them. Additionally, it should be noted that composition functions are mainly similar in different tasks. Therefore, at the end of each training epoch, we fine-tune our shared layer on the Part-of-speech Tagging task (a task that explicitly considers compositional functions) to enrich our feature space with potentially missed compositional properties of the language model. The model is trained on WSJ dataset with a learning rate of 0.001 and a CRF as output layer (*r.f.* experiment 2). The benefits of this compared to the vanilla version is clear under "+Aux" in Table 2.

Using Unlabeled Data: For each mini-batch, the uncertainty regularizer calculates the uncertainty to guide the backpropagation toward features that reduce task uncertainty. Since the regularization term does not rely on data labels, we include the unlabeled data in task uncertainty calculation for each mini-batch. The positive effects are clear in "+Semi" in Table 2 compared to vanilla version.

Performance Evaluation: We perform the multi-task learning on all 16 tasks to compare the task-specific and overall performance of the proposed method. All of the extensions are added to the vanilla version of UFS-MTL, and the final version

Task	LSTM	MT-DNN	MT-CNN	FS-MTL	SP-MTL	SSP-MTL	ASP-MTL	Meta-MTL	UFS-MTL Vanilla	+Fine	+Pre	+Aux	+Semi	All
Books	79.5	82.2	84.5	82.5	81.2	85.3	84.0	87.5	86.9	85.7	85.9	85.5	85.9	87.9
Electronics	80.5	81.7	83.2	85.7	84.7	87.5	86.8	89.5	87.8	89.0	88.0	88.0	87.9	89.8
DVD	81.7	84.2	84.0	83.5	84.0	86.5	85.5	88.0	86.3	87.2	87.1	86.4	86.5	88.4
Kitchen	78.0	80.7	83.2	86.0	85.2	86.5	86.2	91.3	87.8	90.2	89.6	87.9	88.3	91.7
Apparel	83.2	85.0	83.7	84.5	86.5	86.0	87.0	87.0	87.0	89.0	87.2	87.5	86.8	89.0
Camera	85.2	86.2	86.0	86.5	88.0	87.5	89.2	89.7	88.9	89.7	89.0	89.8	89.0	90.0
Health	84.5	85.7	87.2	88.0	87.2	87.5	88.2	90.3	89.3	89.8	89.4	90.0	91.3	90.5
Music	76.7	84.7	83.7	81.2	83.0	85.8	82.5	86.3	83.2	83.9	83.5	84.7	84.1	86.7
Toys	83.2	87.7	89.2	84.5	85.2	87.0	88.0	88.5	87.7	87.9	88.5	87.9	88.1	88.8
Video	81.5	85.0	81.5	83.7	83.2	85.5	84.5	88.3	85.5	87.8	87.7	85.8	86.2	88.7
Baby	84.7	88.0	87.7	88.0	86.7	87.0	88.2	88.0	90.0	90.7	90.3	90.5	91.2	91.2
Magazines	89.2	89.5	87.7	92.5	92.0	88.0	92.2	91.0	92.6	92.7	92.7	92.7	92.8	92.9
Software	84.7	85.7	86.5	86.2	87.0	86.0	87.2	88.5	86.3	86.5	87.4	88.1	88.1	88.7
Sports	81.7	83.2	84.0	85.5	87.2	85.0	85.7	86.7	86.0	86.2	86.3	86.5	86.0	86.8
IMDB	81.7	83.2	86.2	82.5	84.7	84.5	85.5	88.0	84.1	86.8	85.3	84.4	85.4	88.4
MR	72.7	75.5	74.5	74.7	76.0	75.8	76.7	77.0	75.0	75.3	76.9	76.9	77.8	77.9
AVG	81.8	84.3	84.5	84.7	85.1	85.7	86.1	87.9	86.5	87.4	87.2	87.0	87.2	88.6

Table 2: The accuracy of the model on 16 tasks in the dataset (%), compared to its LSTM baseline, and other MTL text classifiers. First, second, and third rankings are denoted in color. Our method (UFS-MTL) performs best in most of the tasks.

("+All") involves all of these improvements on top of the vanilla version.

As can be seen from Table 1, except for two marginal cases, the proposed regularization improves the performance of the FS-MTL up to 5.7% (for Kitchen domain), and 3.5% on average. Besides, this method outperforms other classifiers in most of the tasks and, on average, performs the best. Interestingly, for some of these tasks (such as `Music`, `Toys`, and `Baby`), the LSTM baseline does not perform well, and bag-of-word representation and MLP structure seem more promising. Another interesting pattern is observed when comparing the effect of pre-training and the auxiliary classifier. While both extensions improve the baseline performance, their improvements do not completely stack as they have many commonalities.

Shared Knowledge Transfer: In this study, we strive to provide a better-shared representation between tasks that reduces the uncertainty of all tasks when trained on the data of each of the tasks. We assume that such representation generalizes better on other tasks, and this trained shared layer can be used for other unseen tasks.

To test this hypothesis, we perform a leave-one-out experiment on all of the tasks in which the proposed classifier is trained on the remaining 15 tasks. To test the trained model on the left-out task, we freeze the weights of the shared model, perform 5-fold cross-validation on the target task, and report the result in Table 3. Since only the task-layer of the network for the new task may affect the results, we provide an over parameterized version of our model (UFS-MTL+OP) to ensure that the network can learn the task at hand, given the shared representation. As the table shows, UFS-MTL has a better performance than the FS-MTL, thanks to the uncertainty regularization of the tasks. Also, the effect of over parameterization on the task layer was not considerable on the result, indicating that transferring the trained shared features was the main contributor to the good results of UFS-MTL.

Task	LSTM	SP-MTL	ASP-MTL	Meta-MTL	UFS-MTL	UFS-MTL +OP
ϕ (Books)	79.5	82.2	83.2	86.3	86.4	86.7
ϕ (Electronics)	80.5	84.7	82.2	86.0	86.3	86.6
ϕ (DVD)	81.7	85.2	85.5	86.5	86.4	86.2
ϕ (Kitchen)	78.0	85.0	83.7	86.3	86.7	86.9
ϕ (Apparel)	83.2	85.2	87.5	86.0	88.0	88.2
ϕ (Camera)	85.2	86.7	88.2	87.0	88.2	88.5
ϕ (Health)	84.5	85.5	87.7	88.7	88.9	89.2
ϕ (Music)	76.7	80.0	82.5	85.7	86.7	86.8
ϕ (Toys)	83.2	86.2	87.0	85.3	87.0	87.7
ϕ (Video)	81.5	85.7	85.2	85.5	87.1	87.4
ϕ (Baby)	84.7	83.5	86.5	86.0	86.5	86.7
ϕ (Magazines)	89.2	89.5	91.2	90.3	91.2	91.7
ϕ (Software)	84.7	87.0	85.5	86.5	87.7	87.8
ϕ (Sports)	81.7	83.7	86.7	85.7	86.8	87.4
ϕ (IMDB)	81.7	87.2	87.5	87.3	87.6	88.0
ϕ (MR)	72.7	74.0	75.2	75.5	75.3	75.4
ϕ (AVG)	81.8	84.4	85.3	85.9	85.9	86.0

Table 3: Performance of our model tested on unseen tasks. ϕ(`TASK`) means that we transfer the knowledge of the other 15 tasks to the target `TASK`. Colors show first, second, and third rankings. By learning a shared representation that lowers uncertainty of all tasks while learning from each, we enhanced the overall accuracy of the MTL classifier by 4.1% compared to the baseline.

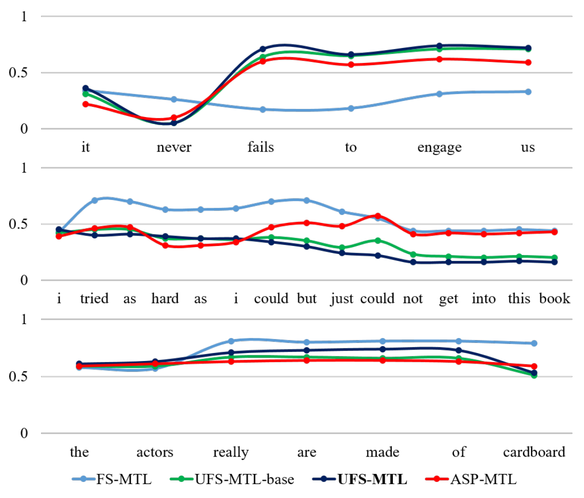


Figure 2: Predicted sentiment score by observing next word. We depict a true positive (top), a true negative (middle) and a false positive case (bottom) of our proposed method (UFS-MTL) compared with FS-MTL, ASP-MTL, and the vanilla version of proposed network, UFS-MTL-base.

Error Analysis: We found two major groups of mistakes made by our model: *(i)* sentences with complicated structures such as complicated forms of negation and *(ii)* sentences that require reasoning or external references (e.g., to pop culture) that conveys a particular sentiment, analogies (e.g., Figure 2 (bottom)) or other types of inferences reaching out of the dataset's scope. In the former case, the use of auxiliary task helps significantly with capturing the essence of the sentences, while the networks that solely focus on sentiment analysis task faced difficulty in capturing the overall sentiment of a complex sentence. In this view, having an auxiliary task to assist the main task such as framework that models definitions of emotions as an auxiliary task while being trained on the primary task of emotion prediction (Singh et al., 2021)) could benefit the model to compensate these errors.

To visualize our model, we picked two successful cases and a failed case of sentiment classification from our model. We depict the sentiment score changes when traversing through words of the sentence by our model and three competing models. It is evident that the uncertainty regularization term guides the network to react to particular words, phrases, and structures considerably. It is also evident that adding auxiliary task (UFS-MTL vs. UFS-MTL-base) boosts the confidence of the method to capture essential structures for the task.

Speed of Convergence: We compared the average loss of the proposed method with Meta-MTL, SP-MTL, and ASP-MTL on train and dev sets of all tasks. We also calculated the total uncertainty of each model on dev. set using (5), for each epoch. As illustrated in Figure 3, our method is more efficient, performs better on dev splits, and reduces the overall task uncertainties more effectively.

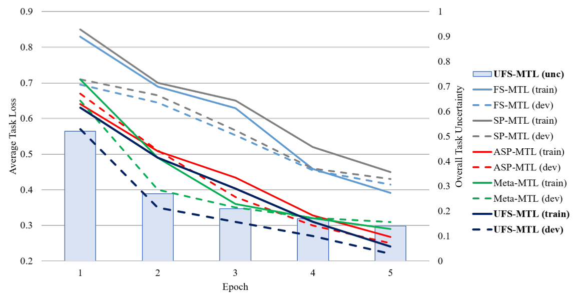


Figure 3: The train and dev loss of several MTL schemes. The overall task uncertainty of UFS-MTL measure by eq(5) on dev data is also shown here.

Effect of Regularization: In this section, we investigate the effect of regularization on the performance of the system. While smaller λ derives the system toward the vanilla FS-MTL, larger λ emphasizes more on the ability of all classifiers to have less uncertain decision criteria. Such a decrease in uncertainty is directly attributed to the shared features since only one of the tasks is trained at a time. Overemphasizing the regularization, on the other hand, pushes the task-specific features in the shared space, as the effect of individual task-specific layers is diminished by increasing the λ. Figure 4 shows the effect of changing λ on the system performance. As larger values of λ prevent the MTL classifier from fine-tuning for each task, the system is prone to catastrophic forgetting resulting from over-generalization of the shared layer in MTL (Subramanian et al., 2018).

Comparing Uncertainty Measures: The choice of uncertainty measure is important to capture the source of uncertainty in the classifier. Table 4 shows the effect of different uncertainty measures on the vanilla UFS-MTL. We denote the average of the softmax outputs of each task used in (Kampffmeyer et al., 2016) by ζ_σ.

While the least confident measure (ζ_{LC}) considers only the most probable class label and tries to maximize it, it effectively throws away information about the remaining label distribution. Entropy in ζ_H considers the full distributions of the posteriors. However, task-specific features in the shared feature space may reduce the entropy for some tasks

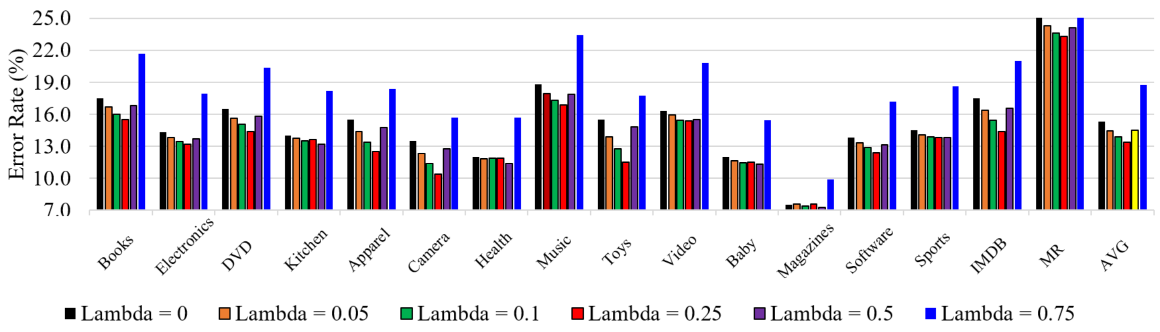


Figure 4: The effect of uncertainty regularization on the UFS-MTL. Small λ reduces the classifier to the FS-MTL, whereas excessively large values of λ prevent fine-tuning for each task in the multi-task learning framework.

	ζ_{LC}	ζ_M	ζ_H	ζ_σ
AVG	85.9	86.5	85.3	85.6

Table 4: Comparing average effect of uncertainty measures on vanilla UFS-MTL performance on all tasks. ζ_{LC} denotes the least confidence uncertainty, ζ_M refers to margin uncertainty, ζ_H indicates the Shanon's entropy, and ζ_σ calculates the average of softmax outputs.

while increasing it for others. Margin uncertainty ζ_M strives to address the shortcoming in the least confident strategy by incorporating the posterior of the second most likely label. Intuitively, instances with large margins have less uncertainty since the second best option is not very competitive.

Discussion on Disentanglement: To obtain an effective shared feature space, while the task-specific features should be pushed out, task-independent features should be pulled in, and redundant features should be punished. PS-MTL explicitly separates the private and shared features, and ASP-MTL tries to push out private features from shared space and omit redundancy by using adversarial training and orthogonality constraint. Yet, there is no encouragement except the training loss to have good shared features in this method. Here, we took an opposite approach and pulled good shared features in shared space (that promote the decisiveness of the MTL) while implicitly pushing away task-specific and redundant features that don't contribute much to overall certainty of MTL.

5.2 Multiple Tasks

In this experiment, we consider a heterogeneous multi-task learning scenario in which three different tasks (part-of-speech tagging, chunking, and named entity recognition) on various datasets are considered. After joint training on all domains (obtaining vanilla version), we include fine-tuning

Datasets	Task	Train	Dev	Test
WSJ	POS Tagging	912,344	131,768	129,654
CoNLL 2000	Chunking	211,727	-	47,377
CoNLL 2003	NER	204,567	51,578	46,666

Table 5: Statistics of 3 datasets for multi-task sequence tagging experiment.

	Chunking (CoNLL2000)	NER (CoNLL2003)	POS Tagging (WSJ)
Single Task Models:			
BiLSTM+CRF	93.67	89.91	97.25
Meta-BiLSTM+CRF	93.71	90.08	97.30
(Collobert et al., 2011)	94.32	89.59	97.29
Multi-Task Models:			
SSP-MTL + CRF	94.32	90.38	97.23
Meta-MTL + CRF	95.11	90.72	97.45
UFS-MTL + CRF **(ours)**	96.11	91.12	97.37

Table 6: Accuracy rates of the models for chunking and NER tasks using F1-score (%) and for POS tagging using Accuracy (%). First, second, and third rankings of each task are denoted in color. Our method (UFS-MTL) outperforms the others in most of the tasks.

and training on unlabeled data to obtain the final UFS-MTL that has the state-of-the-art performance among MTL methods with the same LSTM baseline (Table 6). We excluded pre-training from our model to provide a fair comparison.

Task-Specific Output Layer: Inspired by (Ma and Hovy, 2016), the obtained shared representation is fed to a conditional random field (Lafferty et al., 2001) to perform sequence tagging.

Dataset: For sequence tagging tasks, we use Wall Street Journal (WSJ) subset of Penn Treebank (Marcus et al., 1993), CoNLL 2000 chunking, and CoNLL 2003 English NER datasets (Table 5).

Competitor Models: We compare our method with (Huang et al., 2015) which uses a BiLSTM encoding and CRF output layer. We also compared it with stacked SP-MTL, a bidirectional version of Meta-LSTM (single task), and a Meta-LSTM on top of an SP-MTL, all proposed in (Chen et al., 2018), followed by a CRF output layer. We also compared it with (Collobert et al., 2011).

Results: As shown in Table 6, with the help of uncertainty regularization, we observe that our model is consistently outperforming the competitor models, which shows that our model is very robust and our shared learned features can generalize well among related tasks.

6 Conclusion

In this study, we augment the fully-shared multi-task learning framework with a regularization term to improve the shared representation by lowering the classification uncertainty for all tasks while fine-tuning for each task. The learned representation increased the overall accuracy of the multi-task classifier, achieved competitive results compared to state-of-the-art MTL algorithms, and successfully transferred the knowledge to the unseen tasks.

References

Yaser S Abu-Mostafa. 1990. Learning from hints in neural networks. *Journal of Complexity*, 6(2):192–198.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. 2007. Greedy layer-wise training of deep networks. In *Advances in neural information processing systems*, pages 153–160.

Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. In *ACL'15*, pages 164–169.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 440–447.

Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. 2016. Domain separation networks. In *NIPS'16*, pages 343–351.

R Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias.

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

Junkun Chen, Xipeng Qiu, Pengfei Liu, and Xuanjing Huang. 2018. Meta multi-task learning for sequence modeling. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Kevin Clark, Minh-Thang Luong, Urvashi Khandelwal, Christopher D Manning, and Quoc V Le. 2019. Bam! born-again multi-task networks for natural language understanding. *arXiv preprint arXiv:1907.04829*.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML'08*, pages 160–167. ACM.

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

Li Deng, Geoffrey Hinton, and Brian Kingsbury. 2013. New types of deep neural network learning for speech recognition and related applications: An overview. In *ICASSP'13*, pages 8599–8603. IEEE.

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In *ACL-IJCNLP'15*, pages 845–850.

Tommaso Fornaciari, Alexandra Uma, Silviu Paun, Barbara Plank, Dirk Hovy, and Massimo Poesio. 2021. Beyond black & white: Leveraging annotator disagreement via soft-label multi-task learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2591–2597.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *ICML'15*, pages 1180–1189.

Ross Girshick. 2015. Fast r-cnn. In *ICCV'15*, pages 1440–1448.

Kazuma Hashimoto, Yoshimasa Tsuruoka, Richard Socher, et al. 2017. A joint many-task model: Growing a neural network for multiple nlp tasks. In *EMNLP'17*, pages 1923–1933.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An empirical exploration of recurrent network architectures. In *ICML'15*, pages 2342–2350.

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

Michael Kampffmeyer, Arnt-Borre Salberg, and Robert Jenssen. 2016. Semantic segmentation of small objects and modeling of uncertainty in urban remote sensing images using deep convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 1–9.

Alex Kendall, Vijay Badrinarayanan, and Roberto Cipolla. 2015. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *arXiv preprint arXiv:1511.02680*.

Alex Kendall and Yarin Gal. 2017. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR'18*, pages 7482–7491.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP'14*, pages 1746–1751.

Iasonas Kokkinos. 2017. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6129–6138.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *AAAI'15*.

Gang Liu and Jiabao Guo. 2019. Bidirectional lstm with attention mechanism and convolutional layer for text classification. *Neurocomputing*, 337:325–338.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017a. Adversarial multi-task learning for text classification. In *ACL'17*, pages 1–10.

Sulin Liu, Sinno Jialin Pan, and Qirong Ho. 2017b. Distributed multi-task relationship learning. In *ACM SIGKDD'17*, pages 937–946. ACM.

Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-Yi Wang. 2015. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 912–921.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. *arXiv preprint arXiv:1603.01354*.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pages 142–150. Association for Computational Linguistics.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank.

Elliot Meyerson and Risto Miikkulainen. 2018. Pseudo-task augmentation: From deep multitask learning to intratask sharing—and back. *ICML'18*.

Hyeonseob Nam and Bohyung Han. 2016. Learning multi-domain convolutional neural networks for visual tracking. In *ICPR'16*, pages 4293–4302.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 115–124. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP'14*, pages 1532–1543.

Marek Rei. 2017. Semi-supervised multitask learning for sequence labeling. In *ACL'17*, pages 2121–2130.

Clemens Rosenbaum, Tim Klinger, and Matthew Riemer. 2018. Routing networks: Adaptive selection of non-linear functions for multi-task learning. *ICML'18*.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2017. Sluice networks: Learning what to share between loosely related tasks. *stat*, 1050:23.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *AAAI'19*, volume 33, pages 6949–6956.

Tobias Scheffer, Christian Decomain, and Stefan Wrobel. 2001. Active hidden markov models for information extraction. In *Advances in Intelligent Data Analysis*, pages 309–318, Berlin, Heidelberg. Springer Berlin Heidelberg.

Burr Settles and Mark Craven. 2008. An analysis of active learning strategies for sequence labeling tasks. In *EMNLP'08*, pages 1070–1079. Association for Computational Linguistics.

Gargi Singh, Dhanajit Brahma, Piyush Rai, and Ashutosh Modi. 2021. Fine-grained emotion prediction by modeling emotion definitions. In *2021 9th International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 1–8. IEEE.

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

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *ACL'16*, pages 231–235.

Trevor Standley, Amir R Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. 2019. Which tasks should be learned together in multi-task learning? *arXiv preprint arXiv:1905.07553*.

Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher J Pal. 2018. Learning general purpose distributed sentence representations via large scale multi-task learning. *arXiv preprint arXiv:1804.00079*.

Gajan Suthokumar, Vidhyasaharan Sethu, Kaavya Sriskandaraja, and Eliathamby Ambikairajah. 2020. Adversarial multi-task learning for speaker normalization in replay detection. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6609–6613. IEEE.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

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

Alexandra Uma, Tommaso Fornaciari, Dirk Hovy, Silviu Paun, Barbara Plank, and Massimo Poesio. 2020. A case for soft loss functions. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 8, pages 173–177.

Yongxin Yang and Timothy M Hospedales. 2017. Trace norm regularised deep multi-task learning. *ICLR'2017*.

Jianfei Yu and Jing Jiang. 2016. Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification. In *EMNLP'16*, pages 236–246.

Amir R. Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. 2018. Taskonomy: Disentangling task transfer learning. In *CVPR'18*, pages 3712–3722.

Poorya Zaremoodi, Wray Buntine, and Gholamreza Haffari. 2018. Adaptive knowledge sharing in multi-task learning: Improving low-resource neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 656–661.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS'15*, pages 649–657.