Using task features to Learn Sparse Representations for Zero-shot Transfer in Lifelong Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Using task features to Learn Sparse Representations for Zero-shot Transfer in Lifelong Learning. The title is practically a paragraph by itself, gross!

1 Introduction

In multi-task learning, tasks are often accompanied by metadata. Metadata can be thought of as the description that identifies that task.

As an example take the Landmine dataset. In the Landmine dataset the goal is to correctly classify a region as having or not having a landmine based on various sensor readings. This dataset is separated into tasks corresponding to different geographic regions. Metadata, in this instance, could take the form of geographic information that identifies that tasks.

Consider a second example of motion classification. Given the accelerometer readings at different locations on a person's body, determine the specific action a person was performing (i.e. running, jumping, sitting). This data is separated by individuals performing the experiment where each individual can be treated as a different task. This dataset might have height, weight, age, and gender information that could be used as metadata.

Since this data specifically describes the distinction between tasks we would expect that this information would correlate with the differences in data distributions across tasks. We would like to use this metadata to predict models for new tasks without having to see any data or labels.

2 Setup

Let there be a set of tasks $\boldsymbol{T} = \{T^{(1)}, T^{(2)}, T^{(3)}, \ldots, T^{(\tau)}\}$
Each task consists of a n_t pairs $\{\mathbf{x}_i^{(t)}, \mathbf{y}_i^{(t)}\}$ for $i = 1 \ldots n$ where $\mathbf{x}_i^{(t)}$ is the input vector and $\mathbf{y}_i^{(t)}$ is the target output.
$\mathbf{x}_i^{(t)} \in \mathbb{R}^r$ and $\mathbf{y}_i^{(t)} \in \{0, 1\}$ for classification tasks or $\mathbf{y}_i^{(t)} \in \mathbb{R}$ for regression tasks.

Each task has a corresponding metadata feature vector $\boldsymbol{M} = \{\mathbf{m}^{(1)}, \mathbf{m}^{(2)}, \mathbf{m}^{(3)}, \ldots, \mathbf{m}^{(\tau)}\}$
where the metadeta is dimension l. We similarly let $\mathbf{m}^{(t)} \in \mathbb{R}^l$.
Note that while metadata could be viewed as additional input, it is consistent across a task, and therefore could not be used increase single task performance.

The loss on a specific Task $T^{(t)}$ is

$$\frac{1}{n_t} \sum_{i=1}^{n_t} \mathcal{L}\big(f(\mathbf{x}_i^{(t)}, \Theta^{(t)}), \mathbf{y}_i^{(t)}\big) \tag{1}$$

where $\Theta^{(t)}$ is the parameters for function f, and f is a function that maps $\mathbf{x}_i^{(t)}$ to $\mathbf{y}_i^{(t)}$.

2.1 ELLA framework

Each task is classified by a sparse linear conbination s of a knowledge repository L.

$$L\mathbf{s}^{(t)} = \Theta^{(t)} \tag{2}$$

We denote the set of all sparse task vectors $S = \{\mathbf{s}^{(1)}, \mathbf{s}^{(2)}, \mathbf{s}^{(3)}, \ldots, \mathbf{s}^{(\tau)}\}$

3 Goal

Given a new task $T^{(\tau+1)}$ with unknown sparse representation $\mathbf{s}^{(\tau+1)}$, use the metadata $\mathbf{m}^{(\tau+1)}$ to estimate the unknown $\mathbf{s}^{(\tau+1)}$. This is expressed as the objective function

$$argmin_{\omega} \sum_{t=1}^{\tau} \left(g(\mathbf{m}^{(t)}, \omega) - \mathbf{s}^{(t)}\right)^2 \tag{3}$$

If we assume that $\mathbf{s}^{(i)}$ can be represented as a linear function of $\mathbf{m}^{(i)}$ directly, we get

$$argmin_{W} \sum_{t=1}^{\tau} \left(\mathbf{m}^{(t)}W - \mathbf{s}^{(t)}\right)^2 \tag{4}$$

Where W is an $l \times r$ weight matrix.

4 Similarity and Reconstruction

4.1 Simililarity Measure

In the event that a direct mapping isn't expressive enough, we propose a model based on pairwise similarities. Here we relax the notion that metadata can describe the sparse representation and instead ask that the relation between metadata vectors can be mapped to the relationship between sparse representations.

Let there be a pairwise similarity measure between sparse representations $d(\mathbf{s}^{(i)}, \mathbf{s}^{(j)})$.
The objective function for a mapping h is

$$argmin_{\phi} \sum_{i,j} \left(h\Big(d(\mathbf{m}^{(i)}, \mathbf{m}^{(j)}), \phi\Big) - d(\mathbf{s}^{(i)}, \mathbf{s}^{(j)})\right)^2 \tag{5}$$

For clarity, we make the assumption that the same similarity measure $d()$ is used for both metadata and sparse representations, although this need not be the case. Note that if we believe the similarity function can be learned, then the direct mapping in equation 4 or some non-linear alternative seems sufficient.

4.2 examples of pairwise similarity measures

Example similarity measures include the Pearson correlation coefficient, extended Jaccard coefficient, and the cosine similarity. Most distance measures can also be easily modified into similarity measures for example the Hamming similarity is the count of the elements in common as opposed to the Hamming distance which is the count of the elements that are different. We use the cosine similarity measure as an example as it is particularly well suited to sparse vectors.

4.2.1 cosine similarity example

Defining our pairwise similarity as the cosine similarity we get:

$$d(\mathbf{s}^{(i)}, \mathbf{s}^{(j)}) = \frac{\mathbf{s}^{(i)} \cdot \mathbf{s}^{(j)}}{\|\mathbf{s}^{(i)}\| \cdot \|\mathbf{s}^{(j)}\|} \tag{6}$$

Using sample vectors:

$$\begin{aligned}
\mathbf{s}^{(1)} &= [\quad .2 \quad .4 \quad 0 \quad 0 \quad .1 \quad 0 \quad -.2] \\
\mathbf{s}^{(2)} &= [\quad 0 \quad .2 \quad .1 \quad 0 \quad .1 \quad 0 \quad .1] \\
\mathbf{s}^{(3)} &= [\quad .2 \quad .2 \quad .1 \quad 0 \quad 0 \quad 0 \quad 0]
\end{aligned}$$

we get the following similarity scores

$$\begin{aligned}
d(\mathbf{s}^{(1)}, \mathbf{s}^{(2)}) &= .53 \\
d(\mathbf{s}^{(2)}, \mathbf{s}^{(3)}) &= .63 \\
d(\mathbf{s}^{(1)}, \mathbf{s}^{(3)}) &= .8
\end{aligned}$$

4.3 Nyström Extension

The Nyström Extension applied to Matrices Fowlkes et al. (2004) shows that a symmetric postive definite matrix W can be approximated

$$W = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \approx \begin{bmatrix} A \\ B^T \end{bmatrix} A^{-1} \begin{bmatrix} A & B \end{bmatrix} \tag{7}$$

if we construct a matrix of pairwise similarities we could use the Nyström extension to complete most of the similarities.

Nemtsov also showed that the Nyström extension could be with singular value decomposition on generic UxV matrix as long as W has a square root matrix Nemtsov et al. (2013).

$$W = \begin{bmatrix} A & B \\ F & C \end{bmatrix} \approx \begin{bmatrix} A \\ F \end{bmatrix} A^{-1/2} A^{-1/2} \begin{bmatrix} A & B \end{bmatrix} \tag{8}$$

In which case we could use this reconstruct missing elements of S given we still have some elements already. However we still need to get those elements somehow. ERIC, PLEASE EXPLAIN WHAT YOU HAD IN MIND HERE.

4.4 Reconstruction

Given an appropriate $d()$ and h we can find a $d(\mathbf{s}^{(i)}, \mathbf{s}^{(j)})$ from $\mathbf{m}^{(i)}$, but we don't have $\mathbf{s}^{(i)}$ yet. We need to somehow use all the $d(\mathbf{s}^{(i)}, \mathbf{s}^{(j)})$ to reconstruct the $\mathbf{s}^{(i)}$. To accomplish this we use a function $R()$

$$\begin{aligned}
\boldsymbol{D}_{\mathbf{s}^{(i)}} &= \{d(\mathbf{s}^{(i)}, \mathbf{s}^{(1)}), d(\mathbf{s}^{(i)}, \mathbf{s}^{(2)}), d(\mathbf{s}^{(i)}, \mathbf{s}^{(3)}), \ldots, d(\mathbf{s}^{(i)}, \mathbf{s}^{(\tau)})\} \\
\hat{\mathbf{s}}^{(i)} &= R(\boldsymbol{D}_{\mathbf{s}^{(i)}}, S)
\end{aligned}$$

with the goal that $\hat{\mathbf{s}}^{(\tau+1)} \approx \mathbf{s}^{(\tau+1)}$

Given a function $R()$ we can then use our map h to find $\mathbf{s}^{(i)}$

$$\begin{aligned}
d(\mathbf{s}^{(i)}, \mathbf{s}^{(j)}) &\approx h\Big(d(\mathbf{m}^{(i)}, \mathbf{m}^{(j)}), \phi\Big) \\
\boldsymbol{D}'_{\mathbf{s}^{(i)}} &= \left\{h\Big(d(\mathbf{m}^{(i)}, \mathbf{m}^{(1)}), \phi\Big), h\Big(d(\mathbf{m}^{(i)}, \mathbf{m}^{(2)}), \phi\Big), h\Big(d(\mathbf{m}^{(i)}, \mathbf{m}^{(3)}), \phi\Big), \ldots, h\Big(d(\mathbf{m}^{(i)}, \mathbf{m}^{(\tau)}), \phi\Big)\right\} \\
\hat{\mathbf{s}}^{(i)} &= R(\boldsymbol{D}'_{\mathbf{s}^{(i)}}, S)
\end{aligned}$$

We can make $R()$'s dependence on $\mathbf{m}^{(i)}$ and ϕ more explicit by rewriting the last equation

$$\hat{\mathbf{s}}^{(i)} = R(\mathbf{m}^{(i)}, \phi, S)$$

4.5 Reconstruction Example

When using cosine similarity, one possible choice for $R()$ is the average of $\mathbf{s}^{(j)}$ weighted by the cosine similarity.

$$R(\boldsymbol{D}_{\mathbf{s}^{(i)}}, S) \quad = \quad \frac{\sum_{j=1}^{\tau} d(\mathbf{s}^{(i)}, \mathbf{s}^{(j)}) \mathbf{s}^{(j)}}{\sum_{j=1}^{\tau} d(\mathbf{s}^{(i)}, \mathbf{s}^{(j)})} \tag{9}$$

We again consider the sample vectors and cosine similarity from 4.2.1. Suppose we want to reconstruct $\mathbf{s}^{(3)}$ using only $\mathbf{s}^{(1)}$ and $\mathbf{s}^{(2)}$.

$$R(\boldsymbol{D}_{\mathbf{s}^{(3)}}, S) \quad = \quad \frac{d(\mathbf{s}^{(3)}, \mathbf{s}^{(1)}) \mathbf{s}^{(1)} + d(\mathbf{s}^{(3)}, \mathbf{s}^{(2)}) \mathbf{s}^{(2)}}{d(\mathbf{s}^{(3)}, \mathbf{s}^{(1)}) + d(\mathbf{s}^{(3)}, \mathbf{s}^{(2)})}$$

$$\hat{\mathbf{s}}^{(3)} = [\quad .11 \quad .31 \quad .04 \quad 0 \quad .1 \quad 0 \quad -.07]$$
$$\mathbf{s}^{(3)} = [\quad .2 \quad .2 \quad .1 \quad 0 \quad 0 \quad 0 \quad 0]$$

This however doesn't enforce sparsity in $\hat{\mathbf{s}}^{(3)}$. One possible fix is to take the average number z of non-zero elements in S and limit $\mathbf{s}^{(3)}$ to have at most z elements by zeroing out the smallest elements until $\mathbf{s}^{(3)}$ has at most z non-zero parts. This is hacky, Is there a better way?

4.6 Similarity and Reconstruction Objective Function

$$argmin_{\phi} \sum_{t=1}^{\tau} \left(R(\mathbf{m}^{(i)}, \phi, S) - \mathbf{s}^{(t)} \right)^2 \tag{10}$$

This is incorporated into the lifelong learning framework to ensure that the L and $\mathbf{s}^{(i)}$ are selected in a way where $\mathbf{m}^{(i)}$ can describe (is correlated with) $\mathbf{s}^{(i)}$.

$$\underset{L}{\operatorname{argmin}} \frac{1}{\sqrt{\tau}} \sum_{t=1}^{\tau} \min_{\mathbf{s}^{(t)}, \phi} \Bigg(\underbrace{\frac{1}{n_t} \sum_{i=1}^{n_t} \mathcal{L}\Big(f\big(\mathbf{x}_i^{(t)}; L\mathbf{s}^{(t)}\big), \mathbf{y}_i^{(t)} \Big)}_{\text{task error}} + \underbrace{\mu \|\mathbf{s}^{(t)}\|_1}_{\text{sparsity}} + \underbrace{\left(R(\mathbf{m}^{(i)}, \phi, S) - \mathbf{s}^{(t)} \right)^2}_{\text{metadata reconstruction error}} + \underbrace{\alpha \|\phi\|_1}_{\text{regularization}} \Bigg) + \lambda \|L\|_F^2 \tag{11}$$

citation test: Ruvolo and Eaton (2013)

References

C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the nystrom method. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(2):214–225, 2004.

A. Nemtsov, A. Averbuch, and A. Schclar. Matrix compression using the nystro\" om method. *arXiv preprint arXiv:1305.0203*, 2013.

P. Ruvolo and E. Eaton. ELLA: An efficient lifelong learning algorithm. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28, pages 507–515, 2013. URL `http://machinelearning.wustl.edu/mlpapers/papers/ICML2013_ruvolo13`.