Few Shot Learning

-presented by Shyambhu Mukherjee
For weekly data science seminar

Contents:

What is Few Shot learning?

(1) Traditional machine learning stands for fitting whole and as much as data possible to the machine learning models.
(2) In contrast, humans in learning environment learns with very less no of examples and start showing the learning with enough accuracy.
(3) To depict the same, few shot learning(FSL) or low shot learning is created.
(4) FSL is revolutionary, because FSL means lesser resources and faster results.

Why you should know about it?

In our day to day data science work, we often come across problems, where we can’t get enough data and that creates conflicts. Knowing HOW and WHERE this technique can be applied will help.

Difference between FSL and ZSL

There are a number of terms related to FSL, such as ZSL and OSL which stand for zero shot learning, one shot learning and others. Zero shot learning stands for the models which can perform without any training. So essentially it is about pre-trained models doing different tasks at good enough from start.

OSL stands for one-shot learning, a specific type of FSL.

How does FSL work?

Prior knowledge about similarity: ML models learn patterns in training data that tend to separate different classes even when they are unseen. Traditional ML models can not discriminate classes that are not present in training datasets, whereas few shot learning techniques enable ML models to separate two classes that are not present in the training data and in some applications they can even separate more than two unseen classes

Example: siamese networks, triplet networks, matching networks etc in image classification field.

How does FSL work?

Prior knowledge about learning: ML models use prior knowledge to constrain the learning algorithm to choose parameters that generalize well from few examples.

Examples: LSTMs, Reinforcement learning algorithms, optimization rules, algorithms like MAML, FOMAML etc.

How does FSL work?

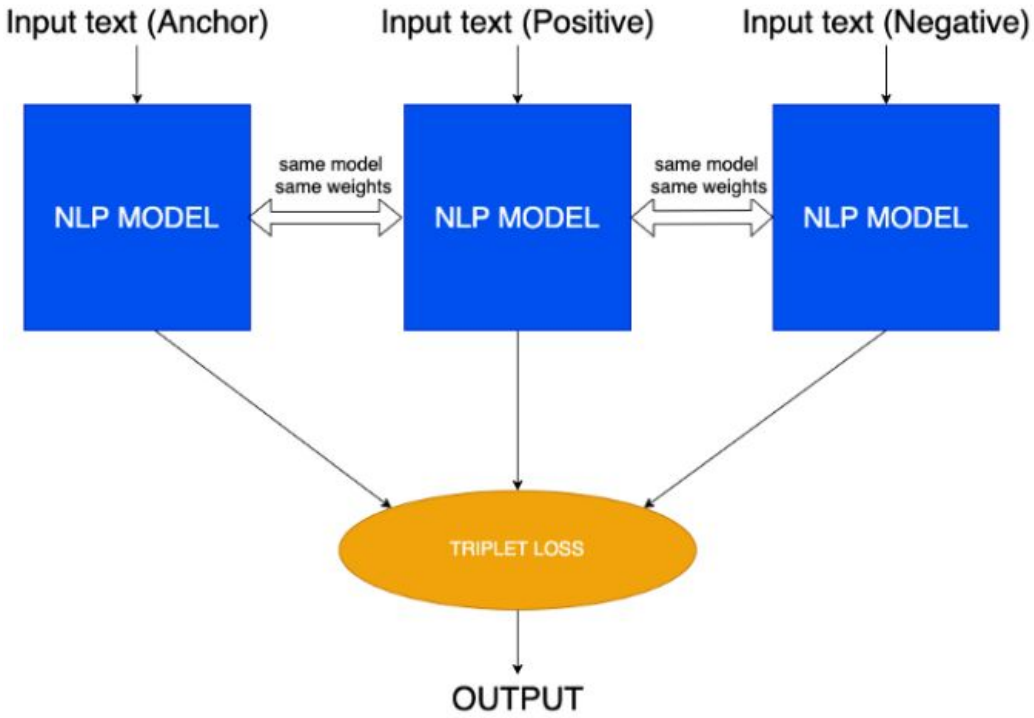
Prior knowledge of data: ML models exploit prior knowledge about the structure and variability of the data, which enables construction of viable models from few examples.

Example: highly pretrained language models like gpt-2,gpt-3 etc.

A deeper look into approaches for NLP

(1) Siamese networks
(2) Triplet loss
(3) Universal sentence encoder

Siamese networks



Siamese Networks with triplet loss

Triplet loss

The triplet loss takes three input embeddings of an anchor, positive and negative data points. The anchor and positive embeddings are of same class and negative embedding is of different class. We try to project the embeddings such that the distance of anchor to negative is alpha more than the distance from anchor to positive. Alpha is also know as the margin, if the difference of distance is greater than the margin than the loss is zero otherwise the difference in distance is considered as the triplet loss and the loss is back-propagated through the siamese network.The mathematical formulation of the loss can be seen below.

$$L(a, p, n) = \frac{1}{N}\left(\sum_{i=1}^{N} \max\{d(a_i, p_i) - d(a_i, n_i) + \text{margin}, 0\}\right)$$

where $d(x_i, y_i) = \|\mathbf{x}_i - \mathbf{y}_i\|_2^2$.

source :https://medium.com/mathematical-beauty/deep-learning-for-cosmetics-2d1b427bbfa2

Universal Sentence Encoder

Model	MR	CR	SUBJ	MPQA	TREC	SST	STS Bench (dev / test)
Sentence & Word Embedding Transfer Learning							
USE_D+DAN (w2v w.e.)	77.11	81.71	93.12	87.01	94.72	82.14	–
USE_D+CNN (w2v w.e.)	78.20	82.04	93.24	85.87	97.67	85.29	–
USE_T+DAN (w2v w.e.)	81.32	86.66	93.90	88.14	95.51	86.62	–
USE_T+CNN (w2v w.e.)	81.18	87.45	93.58	87.32	98.07	86.69	–
Sentence Embedding Transfer Learning							
USE_D	74.45	80.97	92.65	85.38	91.19	77.62	0.763 / 0.719 (r)
USE_T	81.44	87.43	93.87	86.98	92.51	85.38	0.814 / 0.782 (r)
USE_D+DAN (lrn w.e.)	77.57	81.93	92.91	85.97	95.86	83.41	–
USE_D+CNN (lrn w.e.)	78.49	81.49	92.99	85.53	97.71	85.27	–
USE_T+DAN (lrn w.e.)	81.36	86.08	93.66	87.14	96.60	86.24	–
USE_T+CNN (lrn w.e.)	81.59	86.45	93.36	86.85	97.44	87.21	–
Word Embedding Transfer Learning							
DAN (w2v w.e.)	74.75	75.24	90.80	81.25	85.69	80.24	–
CNN (w2v w.e.)	75.10	80.18	90.84	81.38	97.32	83.74	–
Baselines with No Transfer Learning							
DAN (lrn w.e.)	75.97	76.91	89.49	80.93	93.88	81.52	–
CNN (lrn w.e.)	76.39	79.39	91.18	82.20	95.82	84.90	–

Table 2: Model performance on transfer tasks. *USE_T* is the universal sentence encoder (USE) using Transformer. *USE_D* is the universal encoder DAN model. Models tagged with *w2v w.e.* make use of pre-training word2vec skip-gram embeddings for the transfer task model, while models tagged with *lrn w.e.* use randomly initialized word embeddings that are learned only on the transfer task data. Accuracy is reported for all evaluations except STS Bench where we report the Pearson correlation of the similarity scores with human judgments. Pairwise similarity scores are computed directly using the sentence embeddings from the universal sentence encoder as in Eq. (1).

What libraries to use for FSL in nlp?

As you may have already understood, FSL in NLP can be done by using hugely pretrained models, and then applying them on specific downstream tasks. For these, libraries like torchtext, FlairNLP, and huggingface are very useful, with their huge collections of pretrained models and functionality with training them again on new data.

We have not explored the code part, but in the reference we'll leave some links for the same.

What is the difference between transfer learning and FSL?

Few shot learning aims to achieve results with one or very few examples. Imagine an image classification task. You may show an apple and a knife to a human and no further examples are needed to continue classifying. That would be the ideal outcome, but for algorithms.

In order to achieve one-shot learning (or close) we can rely on **knowledge transfer**, just like the human in the example would do .

This brings us to **transfer learning**. Generally speaking, transfer learning is a machine learning paradigm where we train a model on one problem and then try to apply it to a different one.

In the example above, classifying apples and knives is not at all trivial. However, if we are given a neural network that already excels at image classification, with super-human results in over 1000 categories... perhaps it is easy to adapt this model to our specific apples vs knives situation.

Example of an optimization based FSL

Though deep neural networks have shown great success in the large data domain,they generally perform poorly on few-shot learning tasks, where a classifier has to quickly generalize after seeing very few examples from each class. The general belief is that gradient-based optimization in high capacity classifiers requires many iterative steps over many examples to perform well. Here, we propose an LSTM-based meta-learner model to learn the exact optimization algorithm used to train another learner neural network classifier in the few-shot regime. The parametrization of our model allows it to learn appropriate parameter updates specifically for the scenario where a set amount of updates will be made, while also learning a general initialization of the learner (classifier) network that allows for quick convergence of training. We demonstrate that this meta-learning model is competitive with deep metric-learning techniques for few-shot learning.

Full paper link given in references.

Conclusion:

For the last decade, neural network and deep learning culture has climbed to fame on the shoulders of huge datasets. With the onset of techniques like FSL, OSL and ZSL, we can see more complicated learning techniques coming in, which reduces this data dependency. In this presentation we discussed the basics of FSL; and detailed down on how it works in NLP settings. Hope it helped you understand about FSL.

Thanks!

References:

(1) FSL in NLP (this presentation is inspired from this)
(2) what is FSL (this presentation is inspired from this)
(3) Universal sentence encoder
(4) using USE
(5) TARS model using FlairNLP
(6) comparison of FSL, transfer learning
(7) Optimization as FSL technique