On the "Rough Use" of Machine Learning Techniques

Chih-Jen Lin

National Taiwan Univ. MBZUAI

Talk at SIGIR, July 2023

Outline

Outline

Introduction

- Machine learning is everywhere, but unfortunately we are not experts of every method
- Very often we see "inappropriate use" of machine learning techniques
- Examples include
  - reporting training instead of test performance
  - comparing two methods without suitable hyper-parameter searches

Introduction (Cont'd)

- But the reality is that there are more sophisticated examples, for which we broadly call the "rough use" of machine learning techniques
- The setting may be roughly fine, but seriously speaking, is inappropriate
- We briefly discuss two interesting examples

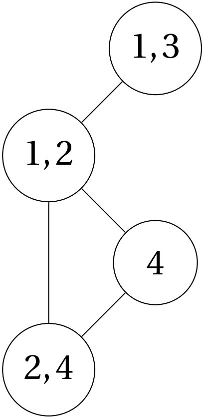
Outline

A Story about Predictions Using Ground Truth

- Predictions using ground truth are impossible in deploying a machine learning model
- But surprisingly unrealistic predictions were used in almost the entire field of graph representation learning
- We reported this story in a paper (Lin et al., 2022)

Graph Representation Learning

- Graph representation learning is a research area to transform a graph into some dense and low dimension embeddings
- This field is quite large, with tens of thousands of papers
- Many use node classification to evaluate the quality of embeddings



Unrealistic Prediction

- A node may have multiple labels: a multi-label classification problem
- The existing prediction process is often as follows
  1. Assumes #associated labels of each test instance is known
  2. Predict this number of labels by selecting those with the largest decision values

Unrealistic Prediction: Example

True labels	Decision values on labels					Prediction	
	1	2	3	4	5	#labels unknown	#labels known
1, 2, 3	0.5	-0.1	0.6	-0.2	-0.5	1, 3	1, 2, 3
4, 5	-0.4	0.2	-0.2	0.6	0.4	2, 4, 5	4, 5
3, 5	-0.7	-0.9	-0.1	-0.4	-0.5		3, 4

- There are five labels; each row is for an instance
- Decision value $\geq 0 \Rightarrow$ has this label; < 0 otherwise

Unrealistic Prediction: Example

True labels	Decision values on labels					Prediction	
	1	2	3	4	5	#labels unknown	#labels known
1, 2, 3	0.5	-0.1	0.6	-0.2	-0.5	1, 3	1, 2, 3
4, 5	-0.4	0.2	-0.2	0.6	0.4	2, 4, 5	4, 5
3, 5	-0.7	-0.9	-0.1	-0.4	-0.5		3, 4

- All decision values are negative
- If # labels is unknown, we will not predict any label

Unrealistic Prediction: Example

True labels	Decision values on labels					Prediction	
	1	2	3	4	5	#labels unknown	#labels known
1, 2, 3	0.5	-0.1	0.6	-0.2	-0.5	1, 3	1, 2, 3
4, 5	-0.4	0.2	-0.2	0.6	0.4	2, 4, 5	4, 5
3, 5	-0.7	-0.9	-0.1	-0.4	-0.5		3, 4

- In the practical use, # labels is unknown
- If # labels is assumed to be known, overestimation tends to occur in evaluation (detailed theory omitted)

Wide Use of Unrealistic Predictions

- People did acknowledge that the setting is unrealistic
- Faerman et al. (2018): "Precisely, this method uses the actual number of labels k each test instance has. [...] In real world applications, it is fairly uncommon that users have such knowledge in advance"

Wide Use of Unrealistic Predictions (Cont'd)

- So why were unrealistic predictions widely used?
- Many papers naturally follow conventions from previous works

  Chanpuriya and Musco (2020): "As in Perozzi et al. (2014) and Qiu et al. (2018), we assume that the number of labels for each test example is given"

Wide Use of Unrealistic Predictions (Cont'd)

- Multi-label classification is considered difficult for researchers in graph-representation learning

  Li et al. (2016): "As the datasets are not only multi-class but also multi-label, we usually need a thresholding method to test the results. But literature gives a negative opinion of arbitrarily choosing thresholding methods"
- We will briefly discuss multi-label classification and explain what the thresholding issue is

Multi-label Classification

- Assume k is the number of labels.
- A simple multi-label method is to assume independence of labels and decompose the problem into k binary sub-problems:

$$f(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \ldots, f_k(\boldsymbol{x}))$$

- Then

$$f_j(\boldsymbol{x}) = \begin{cases} \geq 0 & \text{has label } j \\ < 0 & \text{has not} \end{cases}$$

- The strategy is also known as binary relevance

One-vs-rest (Binary Relevance)

- We learn $f_j(\boldsymbol{x})$ by minimizing

$$\begin{array}{c}\text{training errors of data with label } j \\ + \\ \text{training errors of data without label } j\end{array}$$

- We call this one (data of one label as positive) versus the rest (data of rest labels as negative)

Problems of One-vs-rest

- Macro-F1 results on three graph representation learning methods (larger better)

Training and prediction methods	Macro-F1 DeepWalk	Macro-F1 Node2vec	Macro-F1 LINE
unrealistic	0.304	0.306	0.258
one-vs-rest	0.195	0.191	0.128

- One-vs-rest has significantly worse performance than unrealistic predictions.

Problems of One-vs-rest (Cont'd)

- Because the data set to get $f_j(\boldsymbol{x})$ is often imbalanced, $f_j(\boldsymbol{x})$ tends to predict that $\boldsymbol{x}$ has no label j
- This issue is well known in the area of multi-label classification, and techniques have long been developed to address the issue
- For example, two useful techniques are
  - Thresholding
  - Cost-sensitive (details not shown)

Thresholding Technique

- If
$$f_j(\boldsymbol{x}) \leq 0 \text{ for every test instance } \boldsymbol{x},$$
we can make instances more easily predict label j by considering
$$\Delta_j > 0, \text{ and } f_j(\boldsymbol{x}) \leftarrow f_j(\boldsymbol{x}) + \Delta_j$$
- Δ_j is the threshold value and originally $\Delta_j = 0$

Thresholding Technique (Cont'd)

- We can find suitable Δ_j by a cross-validation procedure (details omitted)
- Such techniques were developed long time ago (Yang, 2001; Lewis et al., 2004; Fan and Lin, 2007)

Thresholding Technique (Cont'd)

- Results

Training and prediction methods	Macro-F1 DeepWalk	Macro-F1 Node2vec	Macro-F1 LINE
unrealistic	0.304	0.306	0.258
one-vs-rest	0.195	0.191	0.128
thresholding	0.299	0.302	0.264

- Thresholding achieves much better results than one-vs-rest

Discussion

- In graph-representation learning, node classification is used to evaluate the quality of embeddings
- In comparing
  1. embedding generation method A and
  2. embedding generation method B,

  the rank by the unrealistic predictions may be the same as that by an appropriate setting
- Then the unrealistic prediction may be fine

Discussion (Cont'd)

- However, the practical deployment can be an issue
- Thus I call this a "rough use" of ML methods: maybe fine in some circumstances, but not appropriate in other situations

Outline

Text Classification in a Recent Study

Chalkidis et al. (2022) released LexGLUE, a collection of legal-document data sets

They report the following Micro-F1 results

Method	ECtHR(A)	ECtHR(B)	SCOTUS	EUR-LEX	LEDGAR	UNFAIR-To
TF-IDF+SVMs	64.5	74.6	78.2	71.3	87.2	95.4
BERT	71.2	79.7	68.3	71.4	87.6	95.6
RoBERTa	69.2	77.3	71.6	71.9	87.9	95.2
DeBERTa	70.0	78.8	71.1	72.1	88.2	95.5
Longformer	69.9	79.4	72.9	71.6	88.2	95.5
BigBird	70.0	78.8	72.8	71.5	87.8	95.7
Legal-BERT	70.0	80.4	76.4	72.1	88.2	96.0
CaseLaw-BERT	69.8	78.8	76.6	70.7	88.3	96.0

Text Classification in a Recent Study (Cont'd)

- Clearly, they aim to compare BERT-based methods, though TF-IDF + linear SVMs is included
- TF-IDF: a bag-of-words way to generate features
- We see TF-IDF + SVMs performs well, especially for the last four problems

Text Classification in a Recent Study (Cont'd)

- In fact, due to the much faster training and smaller model size, in a detailed study (Lin et al., 2023), we show that for document classification, TF-IDF + linear classifiers are a useful baseline
- However, the interesting story I would like to tell is something else
- To begin, for each problem, training, validation and test sets are available
- What was shown is the test performance, independent from training

The Use of Validation Set

- For TF-IDF + linear SVMs, what Chalkidis et al. (2022) did was to
  - combine training and validation sets
  - do cross validation on the combined set to select hyper-parameters
  - re-train the combined set using the best setting
- The purpose of cross validation is to use multiple validation sets for better robustness

The Use of Validation Set (Cont'd)

- But we don't have to do so. For the discussion, let's write a simpler version of what they did
  - check validation performance for selecting hyper-parameters
  - re-train the combined set using the best setting
- For BERT, what they did was
  - check validation performance for selecting the best epoch
  - use the model at the best epoch for prediction

The Use of Validation Set (Cont'd)

- A while after the paper was published, someone[1] wrote:

  "TF-IDF + SVM ... are pretty high, well, I think they have a bias. ... a retraining ... with both training and validation sets combined, while the other Language Models are only fine-tuned with the training set ...ends up in a biased comparison."
- The authors: "that's a great bug finding!"

[1] https://github.com/coastalcph/lex-glue/issues/32

The Use of Validation Set (Cont'd)

- The user: "this bug probably overestimates the TF-IDF+SVM testing scores for all the datasets, as it is using a larger proportion of data"
- The authors: "Cool, I will rerun all of them and update the paper then. Our faith in deep learning can be restored"
- They updated SVM results by using only the training set

The Use of Validation Set (Cont'd)

- The procedure becomes:
  - split training set to sub_training and sub_validation
  - check performance on sub_validation for selecting hyper-parameters
  - re-train the training set using the best setting
- In this way, validation set is totally excluded

The Use of Validation Set (Cont'd)

- They did so because of thinking that "BERT is only fine-tuned with the training set"
- But did BERT really use only the training set?
- No, it did use the validation set
- Recall that BERT checks validation performance to select the best epoch

Training, Validation, and Test Sets

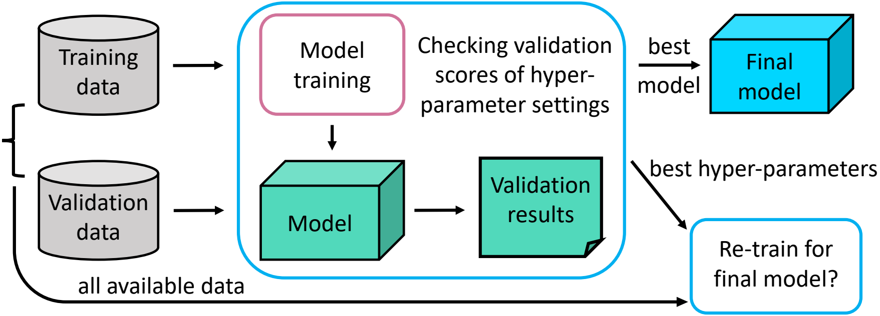
- Let's re-think what training and test mean
- In real world, we are tasked to get a model from some labeled data
- We deploy the model to predict future test data without labels
- Later, labels of test data become available, and we can obtain the test performance
- In an academic study, we use training and test sets to simulate the real scenario

Training, Validation, and Test Sets (Cont'd)

- The test set must not be used in the training process because it represents future unknown data
- However, there is no constraint on how we use the training data $\Rightarrow$ we should do the best to use all labeled data
- Their original way of re-training linear SVMs on the combined (training + validation) set is indeed suitable
- This is a common practice for many classification methods

Re-training or Not

- Thus we see an issue of re-training or not



Re-training or Not (Cont'd)

- But why for BERT they did't train the combined set to get the final model?
- The reason is that for neural networks, usually we rely on validation performance for terminating the optimization process or selecting the best epoch
- Thus we may not be able to use all labeled data for training!
- However, other classification methods may not have this issue

Re-training or Not (Cont'd)

- Consider K-nearest neighbor. Once K is decided, the training process is to save all labeled data as the model
- In this regard, not being able to do easy re-training on all labeled data is a drawback of deep learning
- One shouldn't say that because of this, other classification methods should also exclude some labeled data for obtaining the final model!

Re-training or Not (Cont'd)

- For neural networks, some techniques can be developed so we can do the re-training on all labeled data
- This is an important research issue, though we don't discuss details here
- For these sets, we do hyper-parameter search and re-training for BERT. BERT results improve, though TF-IDF + SVM are still competitive

Re-training or Not (Cont'd)

- For this story, our point here is that people may not think clearly about the relation of training, validation, and test sets
- Then we end up with a rough instead of a rigorous use of machine learning methods

Outline

Seriousness of the Situation?

- The phenomena of rough use of machine learning methods is common and sometimes unavoidable
- The reason is that nothing is called a perfect use of a machine learning method
- One may be an expert on a method, but has only basic knowledge on another

Seriousness of the Situation? (Cont'd)

- We don't think the machine learning use is a $0/1$ question (i.e., right or wrong)
- Instead, it's more like that we have an interval $[0, 1]$, where

  0: extremely inappropriate use

  1: suitable and experienced use
- What we can do is to have a higher score if possible
- But how?

Seriousness of the Situation? (Cont'd)

- One way is to improve the teaching of machine learning. Also we must encourage machine learning users to rigorously take courses
- The other is about software, for which I will address more

The Importance of Software

- We argue that having high quality and easy-to-use software is an important way to improve the practical use of machine learning techniques
- For the first story, if a package with the thresholding technique was available in the beginning, probably the situation is now different
- For the second story, if packages have the re-training mechanism available, then deep learning users can train the combined set for the final model
- We strongly believe that the community should pay more attention on the software development

Conclusions

- The rough use of machine learning methods is common and sometimes unavoidable
- However, improving the practical use is possible and that's what we should try to achieve