Lessons from Developing Machine Learning Algorithms and Systems

Chih-Jen Lin
Department of Computer Science
National Taiwan University

Talk at Nanyang Technological University, January 16, 2017

Outline

Outline

Introduction

- My past research has been on machine learning algorithms and software
- In this talk, I will share our experiences in developing two packages
  - LIBSVM: 2000–now
    A library for support vector machines
  - LIBLINEAR: 2007–now
    A library for large linear classification
- To begin, let me shamelessly describe how these packages have been widely used

Introduction (Cont'd)

- LIBSVM: now probably the most widely used SVM package

  Its implementation document (Chang and Lin, 2011), maintained since 2001 and published in 2011, has been cited more than 29, 000 times (Google Scholar, 11/2016)
- Citeseer (a CS citation indexing system) showed that it's among the 10 most cited work in computer science history

Introduction (Cont'd)

- For LIBLINEAR, it's popularly used in Internet companies

  The official paper (Fan et al., 2008) has been cited more than 4,400 times (Google Scholar, 11/2016)
- Citeseer shows it's second highest cited CS paper published in 2008

Introduction (Cont'd)

- So these past efforts were reasonably successful
- In the rest of this talk I will humbly share some lessons and thoughts from developing these packages

How I Got Started?

- My Ph.D. study was in numerical optimization, a small area not belonging to any big field
- In general, it studies something like

$$\min_{\boldsymbol{w} \in R^n} f(\boldsymbol{w}), \text{ subject to some constraints on } \boldsymbol{w}$$

- After joining a CS department, I found students were not interested in optimization theory
- I happened to find that some machine learning papers must solve optimization problems
- So I thought maybe we can redo some experiments

How I Got Started? (Cont'd)

- While redoing experiments in some published works, surprisingly my students and I had difficulties to reproduce some results
- This doesn't mean that these researchers gave fake results.
- The reason is that as experts in that area, they may not clearly say some subtle steps
- But of course I didn't know because I was new to the area

How I Got Started? (Cont'd)

- For example, assume you have the following data

height	gender
180	1
150	0

  The first feature is in a large range, so some normalization or scaling is needed
- After realizing that others may face similar problems, we felt that software including these subtle steps should be useful

How I Got Started? (Cont'd)

- Lesson: doing something useful to the community should always be our goal as a researcher
- Nowadays we are constantly under the pressure to publish papers or get grants, but we should remember that as a researcher, our real job is to solve problems
- I will further illustrate this point by explaining how we started LIBLINEAR

From LIBSVM to LIBLINEAR

- In 2006, LIBSVM has been popularly used by researchers and engineers
- In that year I went to Yahoo! Research for a 6-month visit
- Engineers there told me that SVM couldn't be applied for their web documents due to lengthy training time
- To explain why that's the case, let's write down the SVM formulation

From LIBSVM to LIBLINEAR (Cont'd)

- Given training data $(y_i, \boldsymbol{x}_i), i = 1, \ldots, l$, $\boldsymbol{x}_i \in R^n, y_i = \pm 1$
- Standard SVM (Boser et al., 1992) solves

$$\min_{\boldsymbol{w}, b} \quad \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \underbrace{C\sum_{i=1}^{l} \max(1 - y_i(\boldsymbol{w}^T\phi(\boldsymbol{x}_i) + b), 0)}_{\text{sum of losses}}$$

- $\boldsymbol{w}^2\boldsymbol{w}/2$: regularization to avoid overfitting
- Decision function

$$\text{predicted label of } \boldsymbol{x} = \text{sgn}(\boldsymbol{w}^T\phi(x) + b)$$

From LIBSVM to LIBLINEAR

- Loss term: we hope

$$y_i \text{ and } \boldsymbol{w}^T\phi(\boldsymbol{x}_i) + b \text{ have the same sign}$$

- $\boldsymbol{x}$ is mapped to a higher (maybe infinite) dimensional space:

$$\boldsymbol{x} \rightarrow \phi(\boldsymbol{x})$$

The reason is that with more features, data are more easily separated

From LIBSVM to LIBLINEAR (Cont'd)

- However, the high dimensionality causes difficulties in training and prediction
- People use kernel trick (Cortes and Vapnik, 1995) so that
$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) \equiv \phi(\boldsymbol{x}_i)^T \phi(\boldsymbol{x}_j)$$
can be easily calculated
- All operations then rely on kernels (details omitted)
- Unfortunately, the training/prediction cost is still very high

From LIBSVM to LIBLINEAR (Cont'd)

- At Yahoo! I found that these document sets have a large number of features
- The reason is that they use the bag-of-words model
  $\Rightarrow$ every English word corresponds to a feature
- After some experiments I realized that if each instance already has so many features
  $\Rightarrow$ then a further mapping may not improve the performance much
- Without mappings, we have linear classification:

$$\phi(\boldsymbol{x}) = \boldsymbol{x}$$

From LIBSVM to LIBLINEAR (Cont'd)

Comparison between linear and kernel

	Linear		RBF Kernel	
Data set	Time	Accuracy	Time	Accuracy
MNIST38	0.1	96.82	38.1	99.70
ijcnn1	1.6	91.81	26.8	98.69
covtype	1.4	76.37	46,695.8	96.11
news20	1.1	96.95	383.2	96.90
real-sim	0.3	97.44	938.3	97.82
yahoo-japan	3.1	92.63	20,955.2	93.31
webspam	25.7	93.35	15,681.8	99.26

Size reasonably large: e.g., yahoo-japan: 140k instances and 830k features

From LIBSVM to LIBLINEAR (Cont'd)

Comparison between linear and kernel

	Linear		RBF Kernel	
Data set	Time	Accuracy	Time	Accuracy
MNIST38	0.1	96.82	38.1	99.70
ijcnn1	1.6	91.81	26.8	98.69
covtype	1.4	76.37	46,695.8	96.11
news20	1.1	96.95	383.2	96.90
real-sim	0.3	97.44	938.3	97.82
yahoo-japan	3.1	92.63	20,955.2	93.31
webspam	25.7	93.35	15,681.8	99.26

Size reasonably large: e.g., yahoo-japan: 140k instances and 830k features

From LIBSVM to LIBLINEAR (Cont'd)

Comparison between linear and kernel

	Linear		RBF Kernel	
Data set	Time	Accuracy	Time	Accuracy
MNIST38	0.1	96.82	38.1	99.70
ijcnn1	1.6	91.81	26.8	98.69
covtype	1.4	76.37	46,695.8	96.11
news20	1.1	96.95	383.2	96.90
real-sim	0.3	97.44	938.3	97.82
yahoo-japan	3.1	92.63	20,955.2	93.31
webspam	25.7	93.35	15,681.8	99.26

Size reasonably large: e.g., yahoo-japan: 140k instances and 830k features

From LIBSVM to LIBLINEAR (Cont'd)

- We see that training time for linear is much shorter
- In the kernel SVM optimization problem, the number of variables (length of $\boldsymbol{w}$ or $\phi(\boldsymbol{x}_i)$) may be infinite
- We cannot directly do the minimization over $\boldsymbol{w}$
- Instead, people rely on a fact that

$$\text{optimal } \boldsymbol{w} = \sum_i y_i \alpha_i \phi(\boldsymbol{x}_i)$$

and do the minimization over $\boldsymbol{\alpha}$

From LIBSVM to LIBLINEAR (Cont'd)

- If we don't do the mapping

  $$\text{optimal } \boldsymbol{w} = \sum_i y_i \alpha_i \boldsymbol{x}_i$$

  is a vector that can be written down
- Then we are able to develop much more efficient optimization algorithms
- After the Yahoo! visit, I spent the next 10 years on this research topic
- This is a good example that I identify the industry needs and bring directions back to school for deep studies

Outline

Most Users aren't ML Experts

- Nowadays people talk about "democratizing AI" or "democratizing machine learning"
- The reasons that not everybody is an ML expert
- Our development of easy-to-use SVM procedures is one of the early attempts to democratize ML

Most Users aren't ML Experts (Cont'd)

For most users, what they hope is

- Prepare training and testing sets
- Run a package and get good results

What we have seen over the years is that

- Users expect good results right after using a method
- If method A doesn't work, they switch to B
- They may inappropriately use most methods they tried

From our experiences, ML packages should provide some simple and automatic/semi-automatic settings for users

Most Users aren't ML Experts (Cont'd)

These settings may not be the best, but easily give users some reasonable results

I will illustrate this point by a procedure we developed for SVM

Easy and Automatic Procedure

- Let's consider a practical example from astroparticle physics

```
1  2.61e+01  5.88e+01  -1.89e-01  1.25e+02
1  5.70e+01  2.21e+02  8.60e-02   1.22e+02
1  1.72e+01  1.73e+02  -1.29e-01  1.25e+02
...
0  2.39e+01  3.89e+01  4.70e-01   1.25e+02
0  2.23e+01  2.26e+01  2.11e-01   1.01e+02
0  1.64e+01  3.92e+01  -9.91e-02  3.24e+01
```

- Training set: 3,089 instances

  Test set: 4,000 instances

Easy and Automatic Procedure (Cont'd)

The story behind this data set

- User:

  `I am using libsvm in a astroparticle physics application .. First, let me congratulate you to a really easy to use and nice package. Unfortunately, it gives me astonishingly bad results...`
- OK. Please send us your data
- I am able to get 97% test accuracy. Is that good enough for you ?
- User:

  `You earned a copy of my PhD thesis`

Easy and Automatic Procedure (Cont'd)

- For this data set, direct training and testing yields

  66.925% test accuracy
- But training accuracy close to 100%
- Overfitting occurs because some features are in large numeric ranges (details not explained here)

Easy and Automatic Procedure (Cont'd)

- A simple solution is to scale each feature to $[0, 1]$

$$\frac{\text{feature value} - \text{min}}{\text{max} - \text{min}}$$

- For this problem, after scaling, test accuracy is increased to 96.15%
- Scaling is a simple and useful step; but many users didn't know it

Easy and Automatic Procedure (Cont'd)

- For SVM and other machine learning methods, users must decide certain parameters
- If SVM with Gaussian (RBF) kernel is used,

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\gamma \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}$$

  then γ (kernel parameter) and C (regularization parameter) must be decided
- Sometimes we need to properly select parameters

  $\Rightarrow$ but users may not be aware of this step

Easy and Automatic Procedure (Cont'd)

After helping many users, we came up with a simple procedure

1. Conduct simple scaling on the data
2. Consider RBF kernel $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\gamma\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}$
3. Use cross-validation to find the best parameter C and γ
4. Use the best C and γ to train the whole training set
5. Test

Easy and Automatic Procedure (Cont'd)

- We proposed this procedure in an "SVM guide" (Hsu et al., 2003) and implemented it in LIBSVM
- This procedure has been tremendously useful.

  Now almost the standard thing to do for SVM beginners
- The guide (never published) was cited more than 4,800 times (Google Scholar, 11/16)
- Lesson: an easy and automatic setting is very important for users

Easy and Automatic Procedure (Cont'd)

- An earlier Google research blog "Lessons learned developing a practical large scale machine learning system" by Simon Tong
- From the blog, "It is perhaps less academically interesting to design an algorithm that is slightly worse in accuracy, but that has greater ease of use and system reliability. However, in our experience, it is very valuable in practice."
- That is, a complicated method with a slightly higher accuracy may not be useful in practice

Users are Our Teachers

- While doing software is sometimes not considered as research, users help to point out many useful directions
- Example: LIBSVM supported only two-class classification in the beginning.
- In standard SVM,

$$\text{label } y = +1 \text{ or } -1,$$

but what if we would like to do digit $(0, \ldots, 9)$ recognition?

Users are Our Teachers (Cont'd)

- At that time I was new to ML. I didn't know how many real-world problems are multi-class
- From many users' requests, we realize the importance of multi-class classification
- LIBSVM is among the first SVM software to handle multi-class data.
- We finished a study on multi-class SVM (Hsu and Lin, 2002) that eventually becomes very highly cited.

  More than 6,300 citations on Google Scholar (11/16)

Users are Our Teachers (Cont'd)

- Another example is probability outputs
- SVM does not directly give

$$P(y = 1|\boldsymbol{x}) \text{ and } P(y = -1|\boldsymbol{x}) = 1 - P(y = 1|\boldsymbol{x})$$

- A popular approach to generate SVM probability outputs is by Platt (2000)
- But it's for two-class situations only

Users are Our Teachers (Cont'd)

- Many users asked about multi-class probability outputs
- Thus we did a study in Wu et al. (2004), which has also been highly cited

  Around 1,500 citations on Google Scholar (11/16)
- Lesson: users help to identify what are useful and what are not.

Optimization and Machine Learning

- I mentioned that a useful ML method doesn't need to be always the best
- However, we still hope to get good performance with efficient training/prediction
- Therefore, in my past research, algorithm development plays an important role
- In particular, I worked a lot on optimization algorithms for ML

Optimization and Machine Learning (Cont'd)

- Many classification methods (e.g., SVM, neural networks) involve optimization problems
- For example, SVM solves

$$\min_{\boldsymbol{w},b} \quad \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i=1}^{l}\max(1-y_i(\boldsymbol{w}^T\phi(\boldsymbol{x}_i)+b),0),$$

- Recall I mentioned that my Ph.D. study was in numerical optimization

Optimization and Machine Learning (Cont'd)

- By the time when I graduated, all the fundamental theory/algorithms for general optimization problems

$$\min_{\boldsymbol{w} \in R^n} f(\boldsymbol{w}), \text{ subject to some constraints on } \boldsymbol{w}$$

  have been done
- That is also one reason I decided to move to ML
- However, notice that $f(\boldsymbol{w})$ of ML problems often has special structures/properties

Optimization and Machine Learning (Cont'd)

- For example, we see the loss term is

$$\max(1 - y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b), 0)$$

- It can be separated to two parts

$$\max(1 - y_i d_i, 0) \tag{1}$$

and

$$d_i = \boldsymbol{w}^T \boldsymbol{x}_i + b \tag{2}$$

- (1) is cheap to calculate, but isn't linear
- (2) is expensive, but is linear

Optimization and Machine Learning (Cont'd)

- In developing LIBSVM and LIBLINEAR, we design suitable optimization methods for these special optimization problems
- Some methods are completely new, but some are modification of general optimization algorithms
- Lesson: even if you are in a mature area, applying techniques to other areas can sometimes lead to breakthroughs
- Scenarios are different. In a different area you might have special structures and properties

Optimization and Machine Learning (Cont'd)

- But what if this specialized $f(\boldsymbol{w})$ turns out to be not useful?

  Everyday new ML methods (and optimization formulas) are being proposed
- That's true. But we don't know beforehand. We simply need to think deeply and take risks. Further, failure is an option

Outline

Software versus Experiment Code

- Many researchers now release experiment code used for their papers

  Reason: experiments can be reproduced
- This is important, but experiment code is different from software
- Experiment code often includes messy scripts for various settings in the paper – useful for reviewers
- Software: for general users

  One or a few reasonable settings with a suitable interface are enough

Software versus Experiment Code (Cont'd)

- Reproducibility different from replicability (Drummond, 2009)

  Replicability: make sure things work on the sets used in the paper

  Reproducibility: ensure that things work in general
- Constant maintenance is essential for software
- One thing I like in developing software is that students learned to be responsible for their work

Software versus Experiment Code (Cont'd)

- Most students think doing research is similar to doing homework when they join the lab
- Once they realized that their work is being used by many people, they have no choice but to be careful
- Example: In the morning of Sunday January 27, 2013, we released a new version of a package.

  By noon some users reported the same bug.

  In the afternoon students and I have worked together to fix the problem.

  We released a new version at that night.

Software versus Experiment Code (Cont'd)

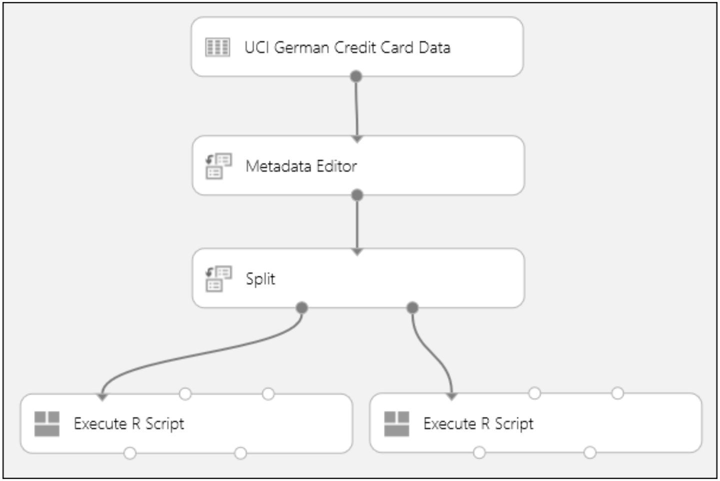
- However, we are researchers rather than software engineers
- How to balance between these maintenance works and the creative research is a challenge
- The community now lacks incentives for researchers to work on high quality software

Interface Issue

- In the early days of open-source ML, most developers are researchers like me
- We propose algorithms as well as design software
- A difference now is that interface becomes much more important
- By interface I mean API, GUI, and others

Interface Issue (Cont'd)

Example: Microsoft Azure ML:



Interface Issue (Cont'd)

- The whole process is by generating a flowchart. Things are run on the cloud
- It makes machine learning easier for non-experts
- However, ML packages become huge
- They can only be either company projects (e.g., TensorFlow, Microsoft Cognitive Toolkit, etc.) or community projects (e.g., scikit-learn, Spark MLlib, etc.)
- A concern is that the role of individual researchers and their algorithm development may become less important

Outline

The Joy of Doing Software

- As a researcher, knowing that people are using your work is a nice experience
- I received emails like

  "It has been very useful for my research"

  "I am a big fan of your software LIBSVM."

  "I read a lot of your papers and use LIBSVM almost daily. (You have become some sort of super hero for me:)) ."

The Joy of Doing Software (Cont'd)

"I would like to thank you for the LIBSVM tool, which I consider to be a great tool for the scientific community."

"Heartily Congratulations to you and your team for this great contribution to the computer society."

"It has been a fundamental tool in both my research and projects throughout the last few years."

Conclusions

- From my experience, developing machine learning software is very interesting and rewarding
- In particular I feel happy when people find my work useful
- We should encourage people in the community to develop high quality machine learning software

Acknowledgments

- All users have greatly helped us to make improvements
- I also thank all my past and current group members