Analysis of a Convex Formulation for Distant Supervision and Fitting a Custom Kernel

Aman Madaan & Suvadeep Hajra

Supervised by: Prof. Ganesh Ramakrishnan
Department of CSE, IIT Bombay

April 2015

Abstract

We present a detailed analysis of the convex formulation presented in Grave 2014. We then present a novel way to use a polynomial kernel with the existing setup while maintaining tractability which *beats* the original implementation on the AUC metric. We conclude by proposing another possible way of using custom kernels using singular value decomposition.

1 Introduction

The convex formulation presented by Grave uses linear kernel. An obvious possible improvement of the existing methods is using different kernels, like the polynomial kernel or one of the kernels used in Culotta and Sorensen [2004], Bunescu and Mooney [2005] or Zelenko et al. [2003]. However, in NLP applications, the feature space is huge (order of million features is very common) and thus naively calculating the kernel matrix is not feasible.

We propose a formulation based on the insight that features that are very common may not encode a lot of information about any particular relation, and features that are rare may be very esoteric to be of any use. We thus prune the features and then project them to a feature space where a kernel would have mapped them. We present strength of this method by using the polynomial kernel as a proof of concept.

We follow the notations and the terminology used by Grave throughout the paper.

2 Analysis

2.1 Derivation of the primal problem

Following Bach and Harchaoui, Grave et al. uses linear classifier $\mathrm{W} \in \mathbb{R}^{D\times(K+1)}$, the squared loss and the squared l_2-norm as the regularizer. Thus, their initial formulation of the primal problem becomes:

$$\begin{aligned} \min_{\mathrm{Y,W}} \quad & \frac{1}{2}||\mathrm{Y} - \mathrm{XW}||_F^2 + \frac{\lambda}{2}||\mathrm{W}||_F^2, \\ \text{s.t.} \quad & \mathrm{Y} \in \{0,1\}^{N\times(K+1)}, \\ & \mathrm{Y1} = 1, \\ & (\mathrm{EY}) \circ \mathrm{S} \geq \mathrm{R}. \end{aligned} \tag{1}$$

where $\mathrm{X} \in \mathbb{R}^{N\times D}$ be the feature matrix representing the relation mention candidates. The closed form solution of the matrix W is given by

$$\mathrm{W} = (\mathrm{X}^T X + \lambda \mathrm{I}_D)^{-1}\mathrm{X}^T\mathrm{Y} \tag{2}$$

Putting this value of W into the objective function of the optimization problem (1), we get the objective function as

$$F(\mathrm{Y}) = \frac{1}{2}\mathrm{tr}(\mathrm{Y}^T(\mathrm{I}_N - \mathrm{X}(\mathrm{X}^T\mathrm{X} + \lambda\mathrm{I}_D)^{-1}\mathrm{X}^T)\mathrm{Y}). \tag{3}$$

Applying the Woodbury matrix identity

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}$$

by taking $A = \mathrm{I}_N$, $U = \mathrm{X}$, $V = \mathrm{X}^T$ and $C^{-1} = \lambda\mathrm{I}_D$, we get

$$\mathrm{I}_N - \mathrm{X}(\mathrm{X}^T\mathrm{X} + \lambda\mathrm{I}_D)^{-1}\mathrm{X}^T = (\mathrm{X}\mathrm{X}^T + \lambda\mathrm{I}_N)^{-1}.$$

Applying this identity in Eq. (3) and relaxing the constraints $\mathrm{Y} \in \{0,1\}^{N\times(K+1)}$ into $\mathrm{Y} \in [0,1]^{N\times(K+1)}$, the following convex quadratic problem is obtained:

$$\begin{aligned} \min_{\mathrm{Y}} \quad & \frac{1}{2}\mathrm{tr}(\mathrm{Y}^T(\mathrm{X}\mathrm{X}^T + \lambda\mathrm{I}_N)^{-1}\mathrm{Y}), \\ \text{s.t.} \quad & \mathrm{Y} \geq 0, \\ & \mathrm{Y}1 = 1, \\ & (\mathrm{E}\mathrm{Y}) \circ \mathrm{S} \geq \mathrm{R}. \end{aligned}$$

Since the inequality constraints might not be feasible, the authors have added the penalized slack variables $\xi \in \mathbb{R}^{I\times(K+1)}$ and obtained the final primal problem:

$$\begin{aligned} \min_{\mathrm{Y},\xi} \quad & \frac{1}{2}\mathrm{tr}(\mathrm{Y}^T(\mathrm{X}\mathrm{X}^T + \lambda\mathrm{I}_N)^{-1}\mathrm{Y}) + \mu||\xi||_1, \\ \text{s.t.} \quad & \mathrm{Y} \geq 0, \quad \xi \geq 0, \\ & \mathrm{Y}1 = 1, \\ & (\mathrm{E}\mathrm{Y}) \circ \mathrm{S} \geq \mathrm{R} - \xi. \end{aligned}$$

The following section describes the dual formulation of the above convex problem.

2.2 Derivation of the dual formulation

By introducing the dual variables $\Lambda \in \mathbb{R}^{I\times(K+1)}$, $\Sigma \in \mathbb{R}^{N\times(K+1)}$, $\Omega \in \mathbb{R}^{I\times(K+1)}$ and $\nu \in \mathbb{R}^N$, such that $\Lambda \geq 0$, $\Sigma \geq 0$ and $\Omega \geq 0$, the Lagrangian of the primal problem can be given by

$$\begin{aligned} L(\mathrm{Y},\xi,\Lambda,\Sigma,\Omega,\nu) = \frac{1}{2}\mathrm{tr}\left(\mathrm{Y}^T(\mathrm{X}\mathrm{X}^T + \lambda\mathrm{I}_N)^{-1}\mathrm{Y}\right) + \mu\sum_{i,k}\xi_{ik} - \mathrm{tr}\left(\Lambda^T((\mathrm{E}\mathrm{Y}) \circ \mathrm{S} - \mathrm{R} + \xi)\right) - \mathrm{tr}\left(\Sigma^T\mathrm{Y}\right) \\ -\mathrm{tr}\left(\Omega^T\xi\right) - \nu^T\left(\mathrm{Y}1 - 1\right). \end{aligned} \tag{4}$$

To obtain the dual function, we need to differentiate the Lagrangian with respect to the primal variables Y and ξ. First differentiating with respect to ξ_{ik} and setting it to zero, we get

$$\frac{\delta L}{\delta \xi_{ik}} = \mu - \Lambda_{ik} - \Omega_{ik} = 0. \tag{5}$$

Applying this equality into Eq. (4), we get the modified Lagrangian,

$$\begin{aligned} \bar{L}(\mathrm{Y},\Lambda,\Sigma,\Omega,\nu) &= \frac{1}{2}\mathrm{tr}\left(\mathrm{Y}^T(\mathrm{X}\mathrm{X}^T + \lambda\mathrm{I}_N)^{-1}\mathrm{Y}\right) - \mathrm{tr}\left(\Lambda^T((\mathrm{E}\mathrm{Y}) \circ \mathrm{S} - \mathrm{R})\right) - \mathrm{tr}\left(\Sigma^T\mathrm{Y}\right) - \nu^T\left(\mathrm{Y}1 - 1\right) \\ &= \frac{1}{2}\mathrm{tr}\left(\mathrm{Y}^T(\mathrm{X}\mathrm{X}^T + \lambda\mathrm{I}_N)^{-1}\mathrm{Y}\right) - \mathrm{tr}\left((\Lambda \circ \mathrm{S})^T\mathrm{E}\mathrm{Y})\right) - \mathrm{tr}\left(\Sigma^T\mathrm{Y}\right) - \mathrm{tr}(1\nu^T\mathrm{Y}) + \mathrm{tr}(\Lambda^T\mathrm{R}) + \nu^T 1. \end{aligned} \tag{6}$$

Differentiating $\bar{L}$ with respect to Y and setting it ot zero, we get

$$\nabla_{\mathrm{Y}}\bar{L} = (\mathrm{X}\mathrm{X}^T + \lambda\mathrm{I}_N)^{-1}\mathrm{Y} - \mathrm{E}^T(\Lambda \circ \mathrm{S}) - \Sigma - \nu 1^T = 0. \tag{7}$$

Rearranging Eq. (7), we get

$$\mathrm{Y} = (\mathrm{XX}^T + \lambda \mathrm{I}_N)(\mathrm{E}^T(\Lambda \circ \mathrm{S}) + \Sigma + \nu 1^T). \tag{8}$$

Putting the optimal value of Y in Eq. (6), we get the dual function as

$$L^*(\Lambda, \Sigma, \Omega, \nu) = -\frac{1}{2}\mathrm{tr}\left(\mathrm{Z}^T\mathrm{QZ}\right) + \mathrm{tr}\left(\Lambda^T\mathrm{R}\right) + \nu^T 1. \tag{9}$$

where

$$\begin{aligned}\mathrm{Q} &= (\mathrm{XX}^T + \lambda \mathrm{I}_N),\\ \mathrm{Z} &= \mathrm{E}^T(\mathrm{S} \circ \Lambda) + \Sigma + \nu 1^T.\end{aligned}$$

Thus the dual problem can be given by

$$\begin{aligned}\max_{\Lambda,\Sigma,\nu} \quad & -\frac{1}{2}\mathrm{tr}\left(\mathrm{Z}^T\mathrm{QZ}\right) + \mathrm{tr}\left(\Lambda^T\mathrm{R}\right) + \nu^T 1\\ \text{s.t.} \quad & \Lambda_{ik} \geq 0, \quad \Sigma_{nk} \geq 0, \quad \Omega_{ik} \geq 0,\\ & \mu - \Lambda_{ik} - \Omega_{ik} = 0, \qquad \forall i, n, k.\end{aligned} \tag{10}$$

2.3 Duality Gap

The primal problem is the usual least squares which is known to have zero duality gap.

2.4 Solving the Objective

The objective is solved using projected gradient descent. The high level idea is to calculate gradient and then project it with respect to the constraints, so that we stay in the feasibility region while minimizing the objective.

3 An Intuitive Interpretation of the Dual

In this section, we present an intuitive interpretation of the dual objective function. The dual function is produced again for reference as follows:

$$\begin{aligned}\max_{\Lambda,\Sigma,\nu} \quad & -\frac{1}{2}\mathrm{tr}\left(\mathrm{Z}^T\mathrm{QZ}\right) + \mathrm{tr}\left(\Lambda^T\mathrm{R}\right) + \nu^T 1\\ \text{s.t.} \quad & \Lambda_{ik} \geq 0, \quad \Sigma_{nk} \geq 0, \quad \Omega_{ik} \geq 0,\\ & \mu - \Lambda_{ik} - \Omega_{ik} = 0, \qquad \forall i, n, k.\end{aligned} \tag{11}$$

The intuition behind the dual is best explained by the following equivalent problem:

$$\begin{aligned}\min_{\Lambda,\Sigma,\nu} \quad & \frac{1}{2}\mathrm{tr}\left(\mathrm{Z}^T\mathrm{QZ}\right) - \mathrm{tr}\left(\Lambda^T\mathrm{R}\right) - \nu^T 1\\ \text{s.t.} \quad & \Lambda_{ik} \geq 0, \quad \Sigma_{nk} \geq 0, \quad \Omega_{ik} \geq 0,\\ & \mu - \Lambda_{ik} - \Omega_{ik} = 0, \qquad \forall i, n, k.\end{aligned} \tag{12}$$

The first term: $\frac{1}{2}\mathrm{tr}\left(\mathrm{Z}^T\mathrm{QZ}\right)$ is made up of the matrices E, a matrix that notes which entity pairs have which mentions, and S, a matrix that that has a 1 for every entity pair-relation that exists in the knowledge base and a -1 for the entity pair-relations that do not exist in the knowledge base.Thus, it is like the relation matrix that penalizes mistakes.

Q is a matrix that is made up of X^TX, the input. In some sense thus, the quadratic is the information about the knowledge that is contained in the data about the different relations.

The second term: $\mathrm{tr}\left(\Lambda^T\mathrm{R}\right)$ is just a matrix that encodes the knowledge base.

With these explanations, we can again look at equation 12 and note that we just want to *minimize the difference between our knowledge of the world, and what the data encodes*

4 Using a custom kernel

4.1 Why is using a custom kernel difficult?

Given the analysis above, we see that the final desideratum, the Y matrix, as well as the intermediate gradient steps depend on $X^T X$, which is of size N x N, where N is the number of instances. It is standard in the distant supervision world to have the number of instances to the order of millions, making direct computation and storage of the kernel matrix infeasible.

4.2 Exploiting Sparsity for Efficient Kernel Calculation

Grave works around the problem of having to compute the complete kernel matrix by using the following facts:

- *The use of Linear Kernel*
- *The average number of features per instance is much less than the total number of instances*

5 Custom Kernel Plugin using Feature Pruning

We start by pruning features appear in all the instances, and those that appear rarely. This approach is already explored in (Goldberg and Elhadad [2008]). The intuition comes from the linguistic nature of the problem; words, phrases and sentence structure that is present across the relations may not help us in learning anything peculiar about a particular relation, which is what we want.

Throwing away some features helps in reducing the width of our instance matrix, which is going to be useful as discussed below.

Algorithm 1 Featuring Pruning and Projection

```
1: Given X, min, max.
2: Remove all the features from X that appear more than max number of times and less than min number
   of times, let the new matrix be X'.
3: For all Rows $x'$ of $X'$, replace x' by a polynomial extension of x', obtained by retaining all the original
   terms, plus the pair wise product.
4: Use Grave with $X'$
```

6 Experiments

6.1 Polynomial Kernel via Feature Pruning

We discuss methodology and results by adopting only the feature pruning method as discussed in section[5]. We try different pruning thresholds and record the completion time and the Area under the curve for all the configuration as discussed in **??**.

6.2 Results

Table 1 shows the AUC and time of completion for different configurations. We note that pruning features that appear less than 5 times, and more than 10000 times beats the standard version on the AUC.

Figure 1 shows the precision recall curves for different configurations, and Figure 2 shows the precision recall curves for the top 4 configurations. From figure 2, we note that the configuration result3_5_10000.tsv beats the base implementation consistently for half the recall range, and also in the overall AUC.

Figure 3a shows that the value of the objective function decreases rapidly as the number of iterations increases. When the number of iterations is sufficiently large, the objective function almost converges and the rate of decrease in its value with the increase in the number of iterations becomes almost zero. Figure 3b plots the cumulative time taken to solve the dual problem as the number of iterations increases. From the figure we see that the elapsed time is almost linear with the number of iterations.

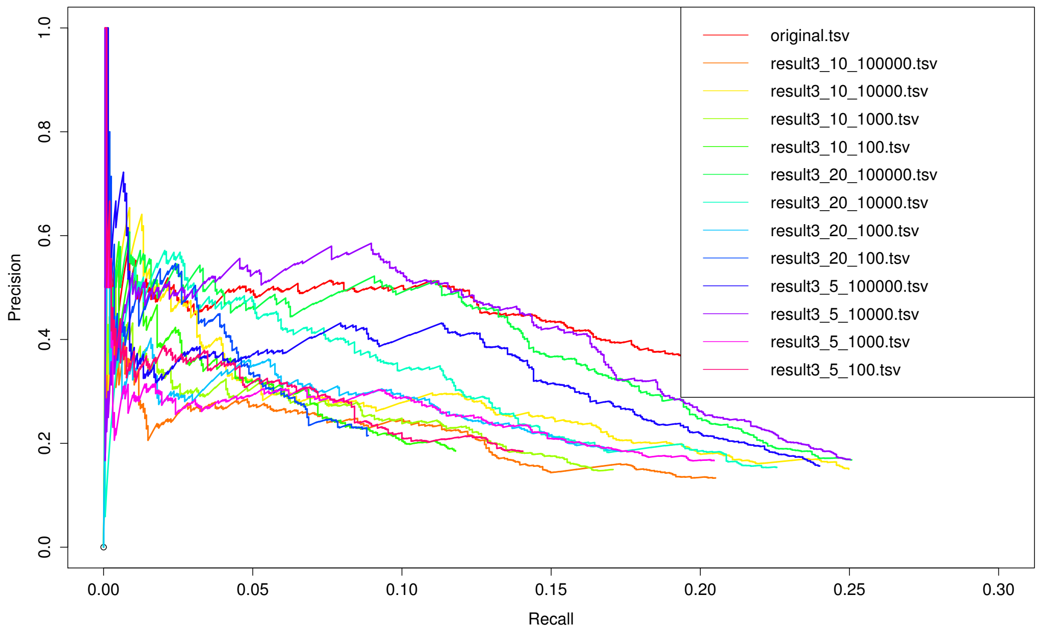


Figure 1: Precision Recall Curve for different Pruning Configurations

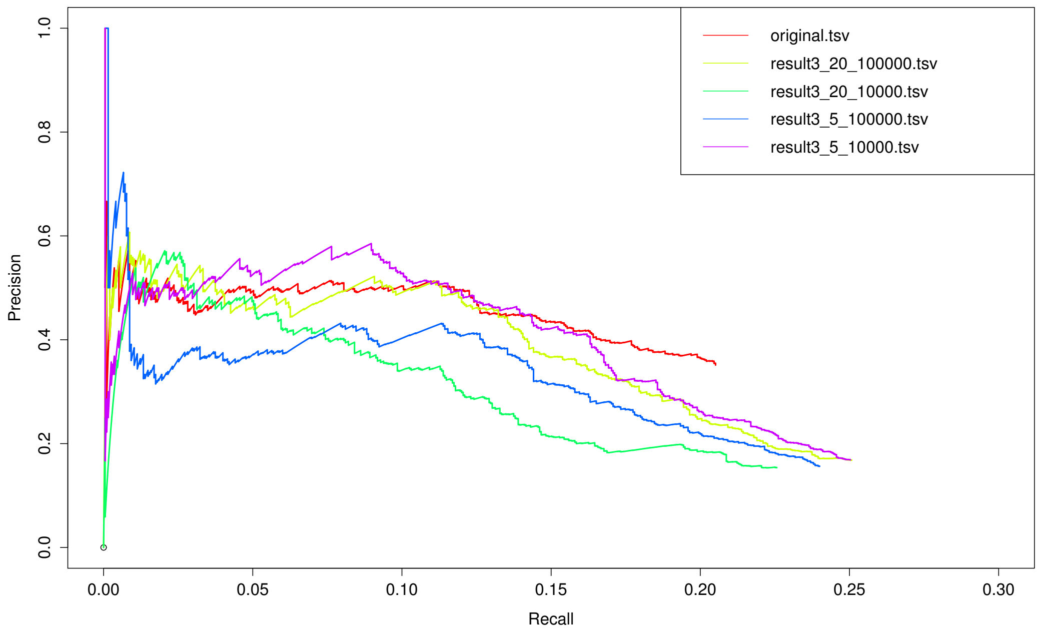
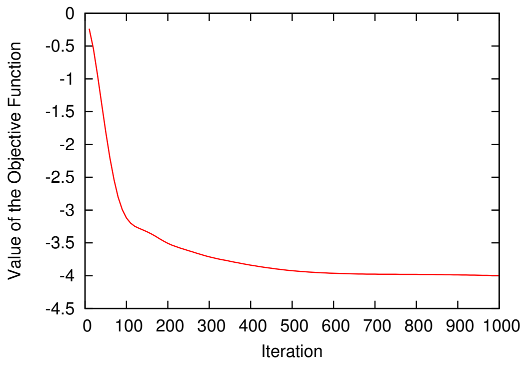


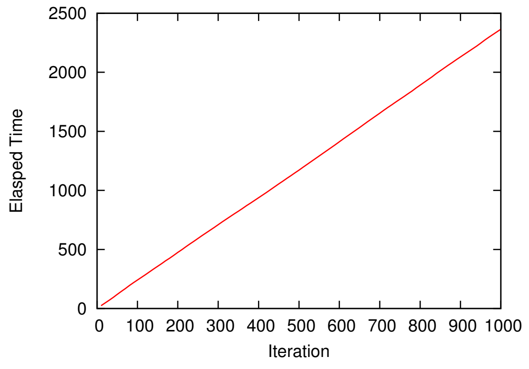
Figure 2: Precision Recall Curve for the top 4 Pruning Configurations

Min Pruning Cutoff	Max Pruning Cutoff	AUC	Time
20	100	0.03311205	211.513000
10	100	0.03736918	283.580000
5	100	0.04061753	386.337000
10	1000	0.04444769	633.979000
10	100000	0.04456001	1369.473000
20	1000	0.04935819	500.905000
5	1000	0.05199495	801.007000
10	10000	0.06930104	1131.839000
20	10000	0.07201145	965.479000
5	100000	0.08066457	1648.713000
No Pruning	No Pruning	0.09428642	551.461000
20	100000	0.09827496	1151.805000
5	10000	0.1033964	1365.037000

Table 1: Results from using Polynomial Kernel via Feature Pruning



(a) The value of the objective function of the dual optimization problem vs. number of iteration.



(b) Elapsed time vs. number of iteration.

Figure 3: The left figure plots how the value of the objective function changes with the increase in number of iterations. The right figure plots the cumulative amounts of time taken to solve the dual optimization problem as the number of iteration increases.

7 Custom Kernel Plugin using Singular Value Decomposition

Let X be the original sparse feature matrix, of dimension N x D where N is the total number of instances and D is the maximum number of features fired for any instance.

Let $K(x_i, x_j)$ be the custom kernel that we want to plugin instead of the existing linear kernel. Then we can find a mapping $\phi : x_i \rightarrow \phi(x_i)$ such that $K(x_i, x_j) = \phi(x_i)^T\phi(x_j)$. Though for some kernels such as Gaussian kernel, $\phi(x_i)$ can be an infinite dimensional vector, we restrict our focus to only those kernels for which the dimension of the projected feature space $\ll N$. There are two ways for getting the feature mapping ϕ from the the kernel K. In one approach, we construct the kernel matrix K of dimension $N \times N$ such that $\mathrm{K}_{ij} = K(x_i, x_j)$. Then, we decompose K as $\Phi\Phi^T$ where Φ is a $N \times D_1$ matrix. Clearly, i^{th} row of Φ represents $\phi(x_i)$. However, since this method needs to compute the $N \times N$ dimensional kernel matrix, it is impractical when N is very large. As an alternative, we can find ϕ analytically from the kernel function K and construct the matrix Φ. For computational efficiency, the column dimension of Φ can be reduce further using Singular Value Decomposition (SVD). This approach is illustrated below:

1. Calculate a matrix Φ of dimensions N x F' where the i^{th} row represents $\phi(x_i)$.
2. Perform a SVD of the matrix Φ, say

$$\Phi = V * \Sigma * U^T \tag{13}$$

3. Construct the F-rank approximation of Φ as

$$\Phi' = \Phi * U_{(:,1:F)} \tag{14}$$

We can now run the code using the new Φ'. The matlab code for the above method is shown in algorithm below.

Algorithm 2 Matlab code

```
1: load data_file
2: sparseMatrix = spconvert(data_file);
3: [U D V] = svds(sparseMatrix, F);
4: projData = sparseNatrix * V;
```

We note that we cannot perform the feature pruning of the type discussed earlier on Φ' because the nature of the features may not be obvious in this case.

References

R. C. Bunescu and R. J. Mooney. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 724–731. Association for Computational Linguistics, 2005.

A. Culotta and J. Sorensen. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 423. Association for Computational Linguistics, 2004.

Y. Goldberg and M. Elhadad. splitsvm: fast, space-efficient, non-heuristic, polynomial kernel computation for nlp applications. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 237–240. Association for Computational Linguistics, 2008.

E. Grave. A convex relaxation for weakly supervised relation extraction.

D. Zelenko, C. Aone, and A. Richardella. Kernel methods for relation extraction. *The Journal of Machine Learning Research*, 3:1083–1106, 2003.