CS57800 Statistical Machine Learning
Homework 3

Andres Bejarano
Department of Computer Science
abejara@purdue.edu

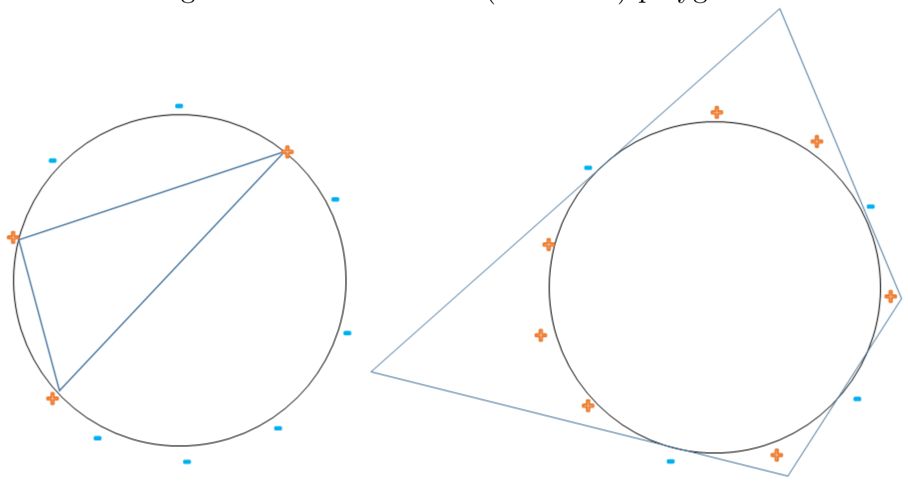
November 10, 2015

1 Foundations

1.1

First let's find the VC dimension for the class of convex polygons with n vertices in the plane. Without loss of generality the vertices of the plane are placed in a circle. Selectively we include any of the vertices in the circle. It is seen two possible situations: More negatives than positives and more positives than negatives. Choosing the points on the circle maximizes the number of possible dichotomies (Figure 1). Then we have that the VC dimension for convex n-sided polygons is $2n+1$.

Figure 1: Convex n-sided (n vertices) polygons



Now we know that the VC dimension for all polygons with n vertices is at least $2n + 1$. Then, for all sets of size $2n+2$ points, there is a labeling of these points that cannot be captured by (even) non-convex polygons with n vertices. Then the VC dimension for all polygons with n vertices is $2n + 2$.

1.2

First we calculate the derivative of the function of the predicted label:

$$\begin{aligned}\hat{y} &= \omega \cdot x + b \\ &= \sum_{i=1}^{D} \omega_i x_i + b\end{aligned}$$

Then:

$$\frac{\partial \hat{y}}{\partial \omega_i} = x_i \qquad \frac{\partial \hat{y}}{\partial b} = 1$$

Now the derivative of the logistic loss function for a fixed value of y applying the chain rule is:

$$\ell(y, \hat{y}) = \frac{1}{\log 2} \log(1 + \exp(-y\hat{y}))$$

$$\frac{\partial \ell(y, \hat{y})}{\partial \hat{y}} = \frac{1}{\log 2} \frac{-y \exp(-y\hat{y})}{1 + \exp(-y\hat{y})} = \frac{-y\sigma(-y\hat{y})}{\log 2}$$

$$\frac{\partial \ell(y, \hat{y})}{\partial \omega_i} = \frac{\partial \ell(y, \hat{y})}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \omega_i} = \frac{-y x_i \sigma(-y\hat{y})}{\log 2}$$

$$\frac{\partial \ell(y, \hat{y})}{\partial b} = \frac{\partial \ell(y, \hat{y})}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial b} = \frac{-y\sigma(-y\hat{y})}{\log 2}$$

Since the above expressions are positive for any fixed $\hat{y} \in [-1, +1]$ then the logistic loss function is convex.

1.3

The training error when a training entry is misclassified is $\frac{1}{n}$. Then it must be guaranteed a training at most $\frac{1}{n}$. Since ϵ_t for the t^{th} iteration is at most γ where $0 < \gamma < 0.5$ then we have that $\epsilon_t < 0.5 - \gamma$.

From class notes we have the following expression for the training error in AdaBoost:

$$H \leq e^{(-2 \sum_t \gamma^2)}$$

Then the error of the final hypothesis is:

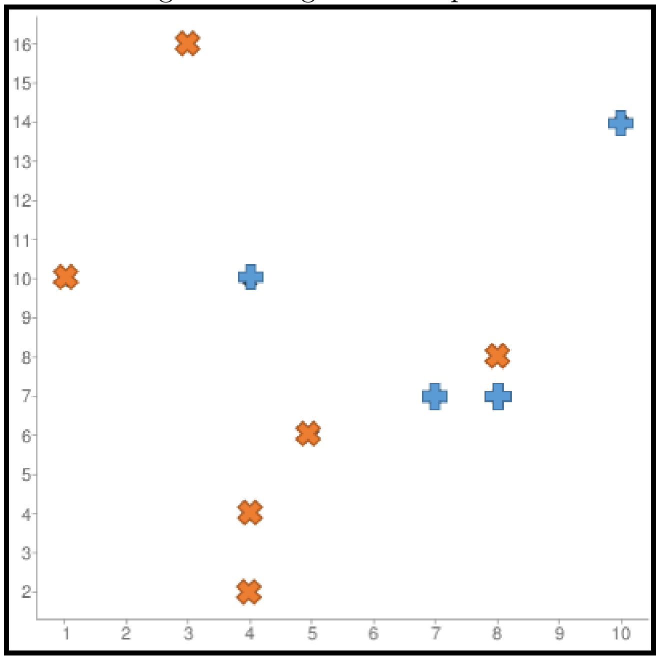
$$\begin{aligned} H &\leq e^{(-2T\gamma^2)} \\ \ln(H) &\leq -2T\gamma^2 \\ \ln(H) &\geq 2T\gamma^2 \\ \frac{\ln(H)}{2\gamma^2} &\leq T \end{aligned}$$

Since H has n training examples we represent that in the expression. Then we got:

$$T \geq \frac{\ln(n)}{2\gamma^2}$$

1.4

Figure 2: Original set of points



Iteration 1:

Weights										
Index	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Probability	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1

- Feature: x_1

- Threshold = 5.7412376
- Target: [0 0 1 0 0 1 1 0 1 0]
- Predicted: [0 0 1 0 0 1 1 0 0 1]
- $\epsilon_1 = 0.2$

- Feature: x_2
  - Threshold = 7.24551758
  - Target: [0 0 1 0 0 1 1 0 1 0]
  - Predicted: [0 0 0 0 0 0 0 0 0 0]
  - $\epsilon_2 = 0.4$

Selected feature: x_1, Threshold: $x_1 < 5.7412376$, beta = 0.25, alpha = 1.38629436112

Iteration 2:

Weights after iteration 1										
Index	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Probability	0.0625	0.0625	0.0625	0.0625	0.0625	0.0625	0.0625	0.0625	0.25	0.25

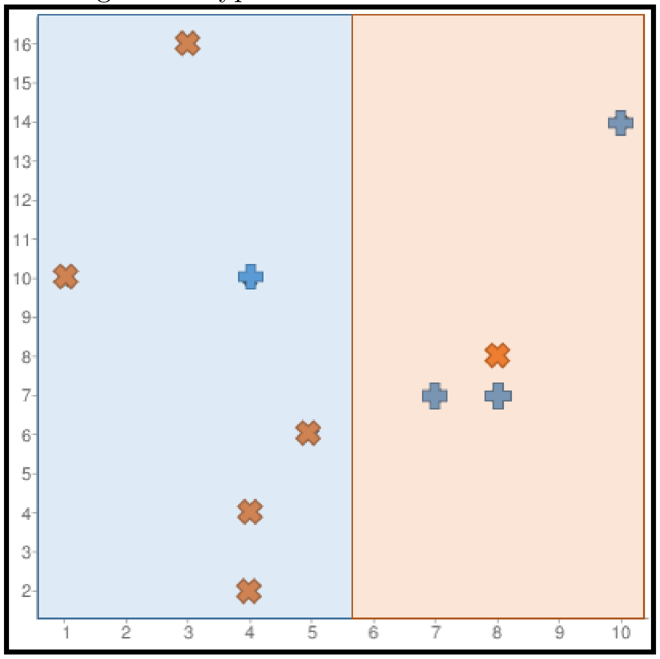
- Feature: x_1
  - Threshold = 5.7412376
  - Target: [0 0 1 0 0 1 1 0 1 0]
  - Predicted: [0 0 1 0 0 1 1 0 0 1]
  - $\epsilon_1 = 0.5$

- Feature: x_2
  - Threshold = 7.24551758
  - Target: [0 0 1 0 0 1 1 0 1 0]
  - Predicted: [0 0 0 0 0 0 0 0 0 0]
  - $\epsilon_2 = 0.4375$

Selected feature: x_2, Threshold: $x_2 < 7.24551758$, beta = 0.777777777778, alpha = 0.251314428281

Weights after iteration 2										
Index	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Probability	0.0555	0.0555	0.0714	0.0555	0.0555	0.0714	0.0714	0.0555	0.2857	0.2222

Figure 3: Hypothesis after two iterations



The final hypothesis after two iterations is:

$$H(x) = (x_1 < 5.75) + 0.25(x_2 < 7.25)$$

1.5

Every kernel K_i has an associated feature map Φ_i and an inner product $\langle\rangle_{H_{K_1}}$. Then, by linearity we have:

$$\alpha K_1(\vec{x},\vec{y}) = \langle\sqrt{\alpha}\Phi_1(\vec{x}), \sqrt{\alpha}\Phi_1(\vec{y})\rangle_{H_{K_1}} \qquad \beta K_2(\vec{x},\vec{y}) = \langle\sqrt{\beta}\Phi_2(\vec{x}), \sqrt{\beta}\Phi_2(\vec{y})\rangle_{H_{K_2}}$$

Then, for $K(\vec{x},\vec{y}) = \alpha K_1(\vec{x},\vec{y}) + \beta K_2(\vec{x},\vec{y})$ we have:

$$\begin{aligned} K(\vec{x},\vec{y}) &= \alpha K_1(\vec{x},\vec{y}) + \beta K_2(\vec{x},\vec{y}) \\ &= \langle\sqrt{\alpha}\Phi_1(\vec{x}), \sqrt{\alpha}\Phi_1(\vec{y})\rangle_{H_{K_1}} + \langle\sqrt{\beta}\Phi_2(\vec{x}), \sqrt{\beta}\Phi_2(\vec{y})\rangle_{H_{K_2}} \\ &= \langle[\sqrt{\alpha}\Phi_1(\vec{x}), \sqrt{\beta}\Phi_2(\vec{x})]\rangle, \langle[\sqrt{\alpha}\Phi_1(\vec{y}), \sqrt{\beta}\Phi_2(\vec{y})]\rangle_{H_{new}} \end{aligned}$$

The last expression indicates that K can be expressed as an inner product. Then, $K(\vec{x},\vec{y}) = \alpha K_1(\vec{x},\vec{y}) + \beta K_2(\vec{x},\vec{y})$ is a valid kernel.

1.6

For SVM, the error for the training entry x_i is defined as:

$$E_i = \frac{1}{2}|y_i - sign(x_i w + b)| \neq 0, \xi_i > 1$$

Given the slack variables in the expression, the relation between the E_i error and the slack variables is expressed as:

$$\implies E = \sum_{i=1}^{M} E_i = \sum_{i=1,\xi_i>1}^{M} E_i \leq \sum_{i=1,\xi_i>1}^{M} \xi_i$$

Then, for the M training entries, the training error is bounded by the sum of the slack variables:

$$\xi_i \geq 0, \forall i = 1, ..., M \implies E \leq \sum_{i=1,\xi_i>1}^{M} \xi_i \leq \sum_{i=1}^{M} \xi_i$$

2 Programming Report

The Sub-Gradient Descent algorithm was implemented using the hinge loss function. All the indicated parameters are considered by the algorithm and changing them alters the training results. A couple of tasks were performed before running the training method:

1. Sentences are cleaned by removing grammatical and grouping symbols.
2. A dictionary of neutral words is implemented in order to improve the classification mechanism. Such neutral words (in neutral.csv) are mostly English prepositions and auxiliary words. It is assumed that such words are useless for classification since they do not affect on the overall sentiment label of the sentence.

The training process runs until the training error is zero or it has run the indicated number of iterations. This algorithm converged faster to a error-free weight vector for the training set.

An important aspect of the experiment was to measure the performance of Gradient Descent using different regularizers. It was observed that using L2 with both unigrams and bigrams feature set has the best performance over the other experiments. The complete set of results for each experiment can be found in the following attahced files: experiment1.txt, experiment2.txt, experiment3.txt, experiment4.txt, experiment5.txt, experiment6.txt.

The following are the parameters and the scored from the executed experiments:

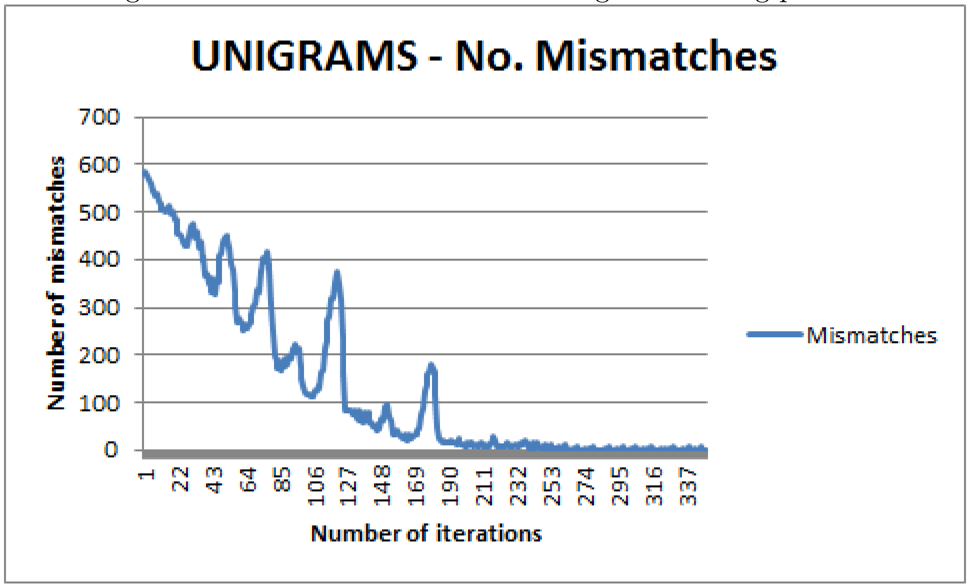
Experiment 1:

Parameter	*Value*
Max Iterations	1000
Regularization	L1
Step Size	0.1
Lambda	0.1
Feature Set	Unigrams

Unigrams Metrics (Before validation)					
Set	*Accuracy*	*Precision*	*Recall*	*Average*	*F-Score*
Training	1.000	1.000	1.000	1.000	1.000
Validating	0.527	1.000	0.715	0.857	0.834
Testing	0.504	1.000	0.710	0.855	0.830

Unigrams Metrics (After validation)					
Set	*Accuracy*	*Precision*	*Recall*	*Average*	*F-Score*
Training	0.883	1.000	0.923	0.961	0.960
Validating	1.000	1.000	1.000	1.000	1.000
Testing	0.508	1.000	0.680	0.840	0.810

Figure 4: Number of mismatches during the training process



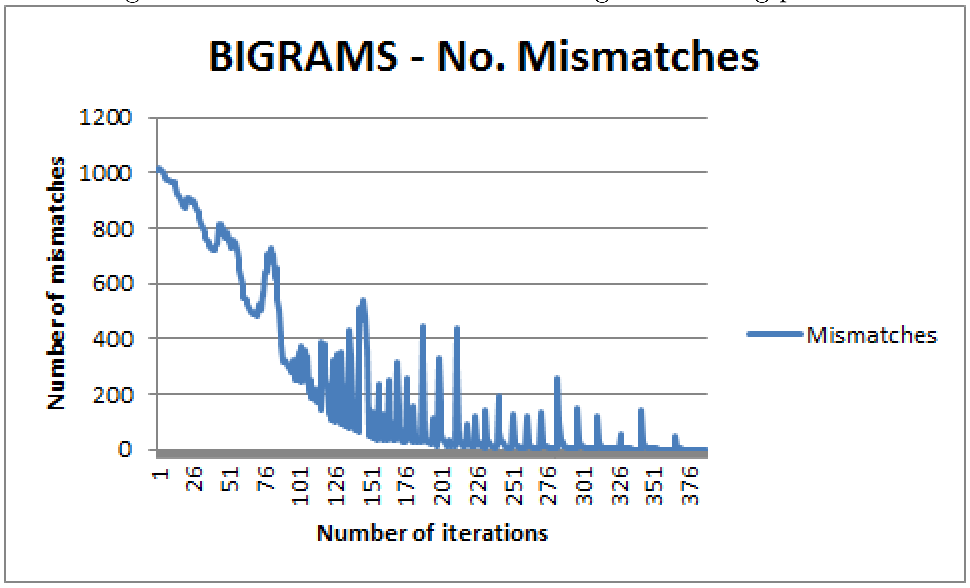
Experiment 2:

Parameter	*Value*
Max Iterations	1000
Regularization	L1
Step Size	0.1
Lambda	0.1
Feature Set	Bigrams

Bigrams Metrics (Before validation)					
Set	*Accuracy*	*Precision*	*Recall*	*Average*	*F-Score*
Training	1.000	1.000	0.999	0.999	0.999
Validating	0.251	1.000	0.367	0.683	0.536
Testing	0.257	1.000	0.369	0.684	0.539

Bigrams Metrics (After validation)					
Set	*Accuracy*	*Precision*	*Recall*	*Average*	*F-Score*
Training	0.876	1.000	0.929	0.964	0.963
Validating	1.000	1.000	0.999	0.999	0.999
Testing	0.264	1.000	0.999	0.999	0.999

Figure 5: Number of mismatches during the training process



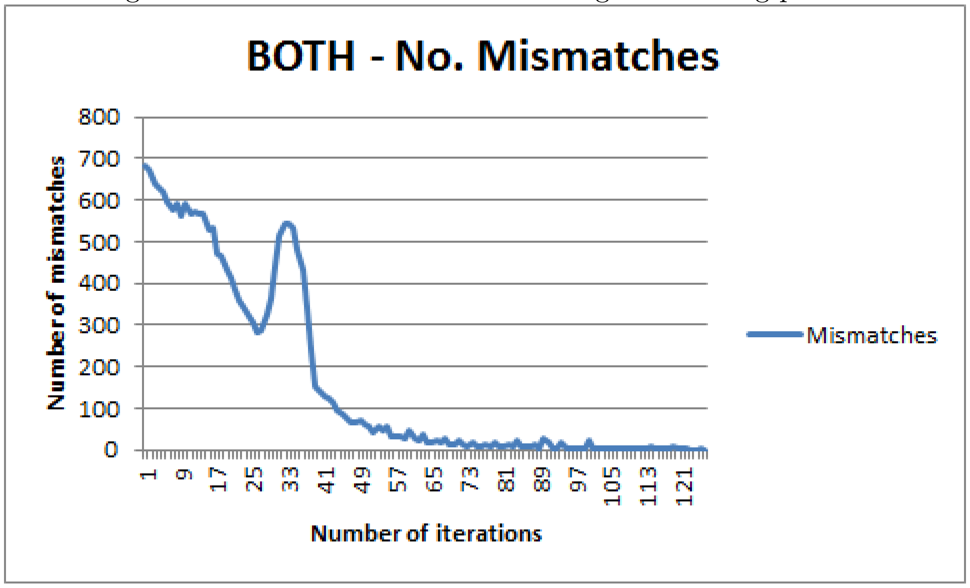
Experiment 3:

Parameter	*Value*
Max Iterations	1000
Regularization	L1
Step Size	0.1
Lambda	0.1
Feature Set	Both

Both Metrics (Before validation)					
Set	*Accuracy*	*Precision*	*Recall*	*Average*	*F-Score*
Training	1.000	1.000	0.999	0.999	0.999
Validating	0.468	1.000	0.602	0.801	0.752
Testing	0.467	1.000	0.616	0.808	0.762

Both Metrics (After validation)					
Set	*Accuracy*	*Precision*	*Recall*	*Average*	*F-Score*
Training	0.948	1.000	0.966	0.983	0.982
Validating	1.000	1.000	0.999	0.999	0.999
Testing	0.484	1.000	0.658	0.829	0.793

Figure 6: Number of mismatches during the training process



Experiment 4:

Parameter	*Value*
Max Iterations	1000
Regularization	L2
Step Size	0.1
Lambda	0.1
Feature Set	Unigrams

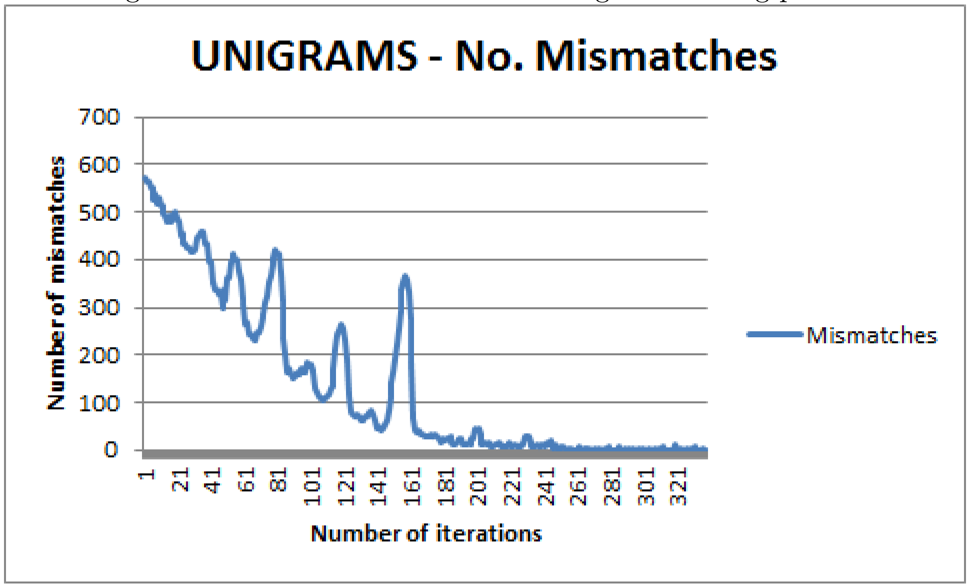
Unigrams Metrics (Before validation)

Set	*Accuracy*	*Precision*	*Recall*	*Average*	*F-Score*
Training	1.000	1.000	1.000	1.000	1.000
Validating	0.533	1.000	0.746	0.873	0.854
Testing	0.502	1.000	0.738	0.869	0.849

Unigrams Metrics (After validation)

Set	*Accuracy*	*Precision*	*Recall*	*Average*	*F-Score*
Training	0.884	1.000	0.930	0.965	0.964
Validating	1.000	1.000	1.000	1.000	1.000
Testing	0.505	1.000	0.704	0.852	0.826

Figure 7: Number of mismatches during the training process



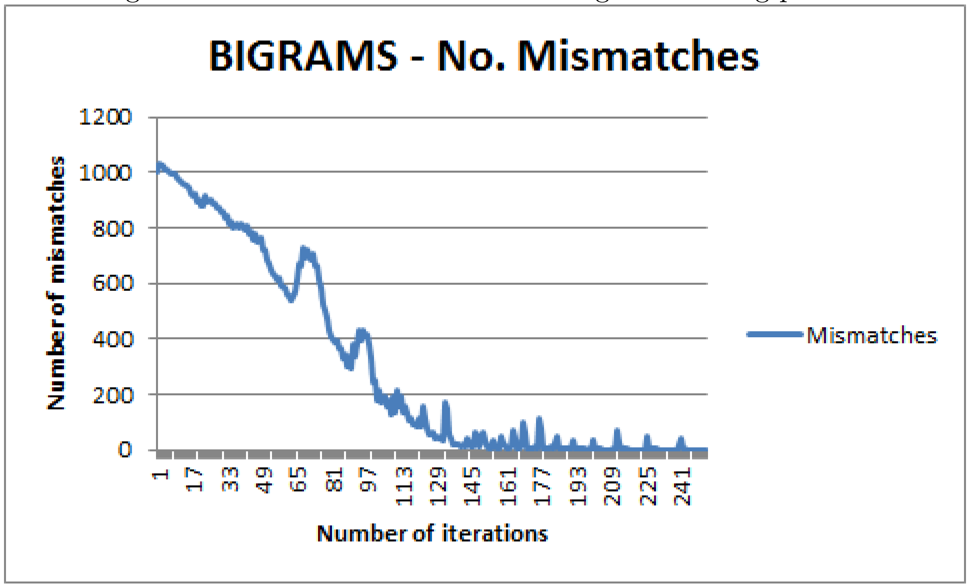
Experiment 5:

Parameter	*Value*
Max Iterations	1000
Regularization	L2
Step Size	0.1
Lambda	0.1
Feature Set	Bigrams

Bigrams Metrics (Before validation)					
Set	*Accuracy*	*Precision*	*Recall*	*Average*	*F-Score*
Training	1.000	1.000	0.999	0.999	0.999
Validating	0.273	1.000	0.456	0.728	0.627
Testing	0.278	1.000	0.459	0.729	0.629

Bigrams Metrics (After validation)					
Set	*Accuracy*	*Precision*	*Recall*	*Average*	*F-Score*
Training	0.936	1.000	0.968	0.984	0.984
Validating	1.000	1.000	0.999	0.999	0.999
Testing	0.294	1.000	0.481	0.740	0.650

Figure 8: Number of mismatches during the training process



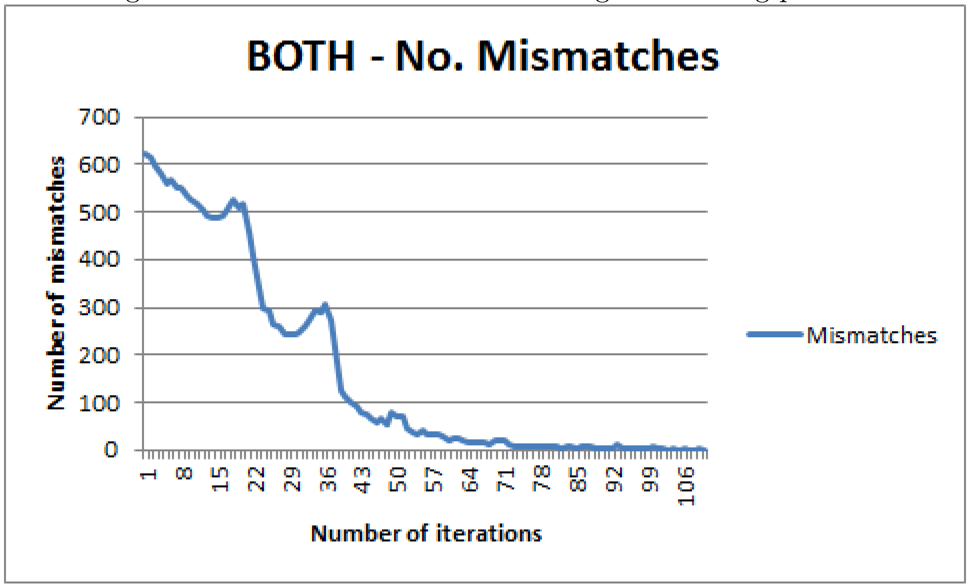
Experiment 6:

Parameter	*Value*
Max Iterations	1000
Regularization	L2
Step Size	0.1
Lambda	0.1
Feature Set	Both

Both Metrics (Before validation)					
Set	*Accuracy*	*Precision*	*Recall*	*Average*	*F-Score*
Training	1.000	1.000	0.999	0.999	0.999
Validating	0.501	1.000	0.714	0.857	0.833
Testing	0.491	1.000	0.721	0.860	0.838

Both Metrics (After validation)					
Set	*Accuracy*	*Precision*	*Recall*	*Average*	*F-Score*
Training	0.964	1.000	0.982	0.991	0.991
Validating	1.000	1.000	0.999	0.999	0.999
Testing	0.497	1.000	0.714	0.857	0.833

Figure 9: Number of mismatches during the training process


BOTH - No. Mismatches
Number of mismatches
Number of iterations
Mismatches