CS224N Programming Assignment 4 Report

Daniel Chia (danchia@stanford.edu), Frank Chen (frankchn@stanford.edu)

We used *three* late days for this assignment. This assignment is submitted both electronically and in paper form on *December 8th, 2012 at 11am*.

1 Introduction

In Programming Assignment 4, we are asked to implement a neural network for named entity recognition. We provide derivations, descriptions of our methods, implementating details, test results and discussion about the pros and cons of our implementation.

Frank wrote the code to load in and preprocess the matrices given while Daniel derived the equations and wrote the code for forward and backpropagation. Frank worked on word vector visualization and the derivation of the equivalence of the softmax classifier and the logistic regression classifier. Frank and Daniel analyzed the results and wrote this report together.

2 Feedforward and Cost Functions

As in the notes, we denote the forward propagation stage, by the following equations, eventually resulting in a prediction $h_\theta(x)$ of whether the word is a PERSON.

$$\begin{aligned} z &= Wx + b_1 \\ a &= f(z) \qquad \text{where } f(x) = \tanh(x) \\ h &= g(U^T a + b_2) \qquad \text{where } g(x) = \frac{1}{1+e^{-x}} \\ J(\theta) &= \frac{1}{m} \sum \left[-y^{(i)}(1 - h_\theta(x^{(i)})) + (1 - y^{(i0)}) h_\theta(x^{(i)}) \right] + \frac{C}{2m} \left[\sum_{j=1}^{nC} \sum_{k=1}^{H} W_{kj}^2 + \sum_{k=1}^{H} U_k^2 \right] \end{aligned}$$

2.1 Gradients

Here, we present the derived gradients used to train the model. Full derivations can be found in the appendix.

$$\begin{aligned} \frac{\partial J(\theta)}{\partial U} &= \frac{1}{m} \sum_{i=1}^{m} \left[\left(-y^{(i)} \frac{1}{h_\theta(x^{(i)})} + (1 - y^{(i)}) \frac{1}{1 - h_\theta(x^{(i)})} \right) h_\theta(x^{(i)})(1 - h_\theta(x^{(i)})) \frac{\partial (U^T a^{(i)} + b_2)}{\partial U} \right] + \frac{C}{m} U \\ &\quad \text{since } g'(x) = g(x)(1 - g(x)) \\ &= \frac{1}{m} \sum_{i=1}^{m} \left[\left(-y^{(i)}(1 - h_\theta(x^{(i)})) + (1 - y^{(i)}) h_\theta(x^{(i)}) \right) a^{(i)} \right] + \frac{C}{m} U \end{aligned}$$

$$\frac{\partial J(\theta)}{\partial b_2} = \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}\frac{1}{h_\theta(x^{(i)})}+(1-y^{(i)})\frac{1}{1-h_\theta(x^{(i)})}\right)h_\theta(x^{(i)})(1-h_\theta(x^{(i)}))\right]$$

$$\begin{aligned}\frac{\partial J(\theta)}{\partial W_{kj}} &= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}(1-h_\theta(x^{(i)}))+(1-y^{(i)})h_\theta(x^{(i)})\right)\frac{\partial(U^T a^{(i)}+b_2)}{\partial W_{kj}}\right]+\frac{C}{2m}(2W_{kj})\\ &= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}(1-h_\theta(x^{(i)}))+(1-y^{(i)})h_\theta(x^{(i)})\right)U_k f'(z_k^{(i)})\frac{\partial(W_{k\cdot}x^{(i)}+(b_1)_k)}{\partial W_{kj}}\right]+\frac{C}{m}W_{kj}\\ &= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}(1-h_\theta(x^{(i)}))+(1-y^{(i)})h_\theta(x^{(i)})\right)U_k(1-\tanh^2 z_k^{(i)})x_j^{(i)}\right]+\frac{C}{m}W_{kj}\end{aligned}$$

$$\begin{aligned}\text{Let } \delta_k^{(i)} &= \left(-y^{(i)}(1-h_\theta(x^{(i)}))+(1-y^{(i)})h_\theta(x^{(i)})\right)U_k(1-\tanh^2 z_k^{(i)})\\ \frac{\partial J(\theta)}{\partial W} &= \frac{1}{m}\sum_{i=1}^{m}\delta^{(i)}\left(x^{(i)}\right)^T+\frac{C}{m}W\frac{\partial J(\theta)}{\partial(b_1)_i}\\ &\quad \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}(1-h_\theta(x^{(i)}))+(1-y^{(i)})h_\theta(x^{(i)})\right)U_k f'(z_k^{(i)})\frac{\partial(W_{k\cdot}x^{(i)}+(b_1)_k)}{\partial W_{kj}}\right]\\ &\quad \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}(1-h_\theta(x^{(i)}))+(1-y^{(i)})h_\theta(x^{(i)})\right)U_k f'(z_k^{(i)})\right]\end{aligned}$$

$$\frac{\partial J(\theta)}{\partial b_1} = \frac{1}{m}\sum_{i=1}^{m}\delta^{(i)}$$

$$\begin{aligned}\frac{\partial J(\theta)}{\partial L_k} &= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}(1-h_\theta(x^{(i)}))+(1-y^{(i)})h_\theta(x^{(i)})\right)\frac{\partial(U^T a^{(i)}+b_2)}{\partial L_k}\right]\\ &= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}(1-h_\theta(x^{(i)}))+(1-y^{(i)})h_\theta(x^{(i)})\right)\left(\sum_{j=1}^{H}U_j f'(z_j^{(i)})\frac{\partial(W_{j\cdot}x^{(i)}+(b_1)_j)}{\partial L_k}\right)\right]\\ &= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}(1-h_\theta(x^{(i)}))+(1-y^{(i)})h_\theta(x^{(i)})\right)\left(\sum_{j=1}^{H}U_j f'(z_j^{(i)})W_{jk}\right)\right]\end{aligned}$$

$$\frac{\partial J(\theta)}{\partial L} = \frac{1}{m}\sum_{i=1}^{m}W^T\delta^{(i)}$$

3 SGD Training and Implementation

We translated and implemented the gradients and SGD in a straight-forward manner in Java. However, we imporved on the given algorithm slightly, by implementing a learning rate schedule. Learning rate is a tradeoff between speed (larger learning rate will give you faster convergence), and precision at the end (too large learning rate will oscillate around minimum). We use

$$learning\,rate = \frac{\alpha}{\beta + iteration}$$

where α, β are parameters that need to be set. This allows us to use a bigger learning rate at the start, which decays as training progresses. A smaller β results in a more aggressive decay.

3.1 Results

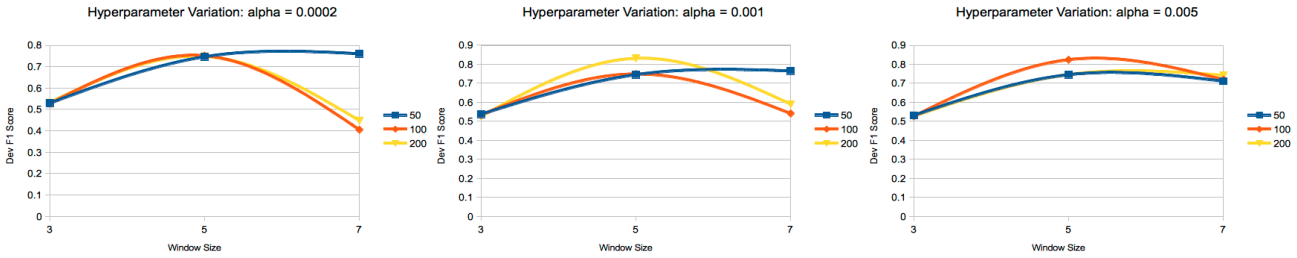
The results as evaluated using the `conlleval.pl` on the `dev` dataset is as follows:

Tag	Precision	Recall	F1
PERSON	56.20%	61.84%	58.89%

4 Network Analysis

4.1 Hyperparameter Variation

Note: Results below show token-based F1 scores.



We have used token-based development F1-scores to tune our hyperparameters rather than entity-level evaluation and we varied the window size from 3 to 7 in steps of 2, number of neurons in the hidden layer between 50 and 200, and the learning rate α between 0.0002 and 0.005. We have plotted the results of the graphs as above. All our training is done with $C = 20$ (i.e. 20 iterations).

From our graph, we can see that performance on the dev-test decreases when the window size is increased beyond 5 uness either $\alpha = 0.0002$ or the size of the hidden layer is 50. We observed a similar trend in the F1 scores on our training set as well.

We believe this may be due to overfitting to the training set. As our data size is rather small, the increase in our learning rates and the number of hidden units will cause our data to conform far too closely to the training set and thus perform rather poorly on the test set. This could perhaps have been fixed with more aggressive regularization by increasing weight decay.

At every iteration we output the value of the objective function. Based on this, we believe we run sufficient iterations of training. However, to be really sure, what we could have done is to validate the trained model every so often and plot the results to verify that additional training does not result in better test scores.

4.2 Error Analysis

4.2.1 Misclassification of `O` as `PERSON`

Place Names and other Proper Nouns There were significant instances of misclassification of proper nouns, such as "National Observatory", "Korea" and "Mercedes-Benz". These words and phrases often have the same characteristics (including the frequency of appearance and capitalization) and appear in the same contexts as phrases refering to individuals. This will throw our algorithms off.

Dates and Numbers We are also classifying dates and numbers (e.g. 1996-08-31, 3, 2) as person entities. We could easily remedy this by forcing the classification of any entity with numbers in them as `O` instead of `PERSON`.

Pronouns Finally, classification of "I", "He" or "She" as persons are also common. We believe that as these pronouns often appear in the same location as people names, the algorithm would be confused. An easy fix for this would merely be adding the list of proper nouns and forcing the classification of any proper noun detected as "O" rather than "PERSON."

4.2.2 Misclassification of `PERSON` as `O`

Non-English Names Our system often misclassifies non-English first and last names as `O` rather than `PERSON`. We believe this is because we have not seen the word in the wordVector or the training data before, and we have to use the UNK token as a substitute for the word itself and only context to figure out whether the current word is a `PERSON` reference or not. One way to provide additional features to the classifier would be to augment the word vector with a feature describing the capitilization of the word (such as all capitals, first capital, last capital).

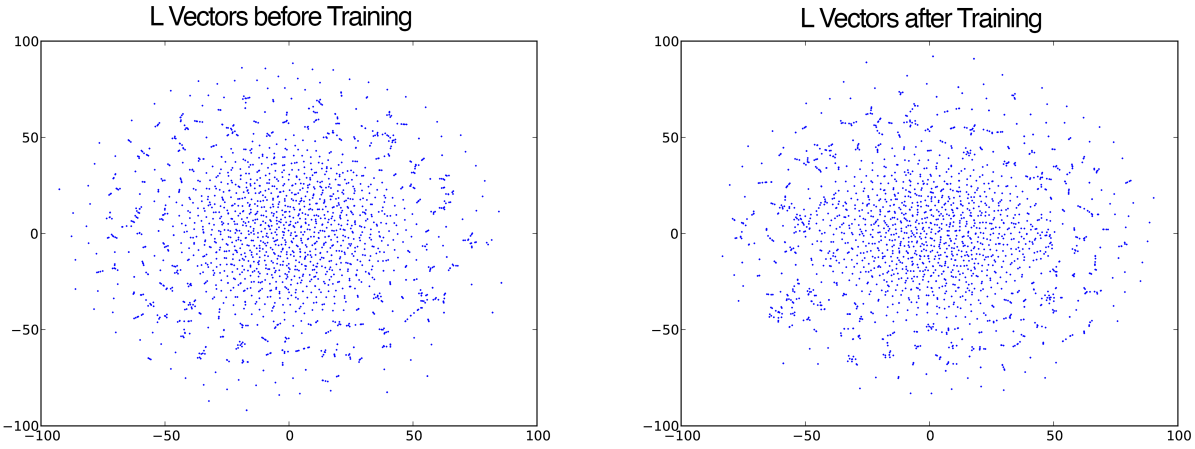
Usage of Only Last Names In some instances (e.g. ...in Gulf await *Clinton* order..., *Wang* was jailed for....), our neural network did not manage to identify the single token as a person entity. We hypothesize this is due to the context of the surrounding tokens. These words may only have activations which cross the threshold when certain other words (e.g. other similar names – such as Bill Clinton) are in the same context as it.

Shortened Names (e.g. P., J.) Shortened names such as P. and J. is not correctly classified as part of person. We believe that this is because the tokens that form these shortened names (e.g. "P.", "W.", etc...) are also part of acronyms in our training text that are not referring to persons. This overloaded usage of the token may well throw our simple classifier off, especially when the training data does not contain any shortened names with that specific token.

4.2.3 Classifying only parts of Entities

Classification of Titles The algorithm believed that the words "Cardinal" and "General" in "Cardinal Wolsey" and "General Kutuzov" are part of the named entity as well. We believe that the correctness of our results versus the gold solution is debatable. We believe that marking "Cardinal Wolsey" as an entity rather than just "Wolsey" is equally correct since "Cardinal Wolsey" refers to a person.

4.3 Word Vector Visualization



We have also visualized a random sampling of L vectors before and after training using the t-SNE algorithm, as suggested by the assignment handout.

The L vectors before and after training does not seem to have much difference at first glance. However, we note that the center area is smaller in the L vector after training, and there are more pronounced clusters in the L-vectors after training than before. The center cluster of individual vectors were also smaller than before training. We believe additional clustering is observed because those words often appear in the context window together in our training set and thus would be grouped closer to each other after our training is done than the starter L vectors given in the starter code.

In addition, we also observed about 90% of L vectors which were unchanged during training as those words did not exist in the training set and thus training on our current dataset would not be useful. We believe that a larger dataset would definitely be more useful for more coverage of all the L vectors and improve the performance of our neural network.

4.4 Softmax Classifier Equivalence to Logistic Regression Classifier

Given a softmax classifier with $k = 2$, the classifier will output the following[1]:

$$\begin{aligned} h_\theta(x) &= E[T(y)|x;\theta] \\ &= \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} \\ &= \begin{bmatrix} \frac{\exp(\theta_1^T x)}{\exp(\theta_1^T x)+\exp(\theta_2^T x)} \\ \frac{\exp(\theta_2^T x)}{\exp(\theta_1^T x)+\exp(\theta_2^T x)} \end{bmatrix} \end{aligned}$$

We note that one of θ_1 and θ_2 is redundant and that we can subtract θ_2 from all the parameters, and letting $\theta' = \theta_1 - \theta_2$ (since $\theta_2 - \theta_2 = 0$), we thus have:

$$\begin{aligned} h_\theta(x) &= \begin{bmatrix} \frac{\exp((\theta_1-\theta_2)^T x)}{\exp((\theta_1-\theta_2)^T x)+1} \\ \frac{1}{\exp((\theta_1-\theta_2)^T x)+1} \end{bmatrix} \\ &= \begin{bmatrix} \frac{\exp(\theta'^T x)}{\exp(\theta'^T x)+1} \\ \frac{1}{\exp(\theta'^T x)+1} \end{bmatrix} \end{aligned}$$

[1] CS229 Lecture Notes 1 Page 26

$$= \begin{bmatrix} 1 - \frac{1}{\exp(\theta'^T x)+1} \\ \frac{1}{\exp(\theta'^T x)+1} \end{bmatrix}$$

We can now observe that the output of the one of the two classes is exactly equivalent to the predictions of the logistic regression hypothesis function[2]: $h_\theta(x) = g(\theta^T x) = \frac{1}{1+\exp(-\theta^T x)}$ when $\theta' = \theta$. The other class is obviously $1 - h_\theta(x)$. Therefore, logistic regression is a special case of softmax regression where $k = 2$.

A Gradient Derivations

$$\begin{aligned}
z &= Wx + b_1 \\
a &= f(z) \qquad \text{where } f(x) = \tanh(x) \\
h &= g(U^T a + b_2) \qquad \text{where } g(x) = \frac{1}{1+e^{-x}} \\
J(\theta) &= \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i0)})h_\theta(x^{(i)})\right] + \frac{C}{2m}\left[\sum_{j=1}^{nC}\sum_{k=1}^{H} W_{kj}^2 + \sum_{k=1}^{H} U_k^2\right] \\
\frac{\partial J(\theta)}{\partial U} &= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}\frac{1}{h_\theta(x^{(i)})} + (1-y^{(i)})\frac{1}{1-h_\theta(x^{(i)})}\right)\frac{\partial h_\theta(x^{(i)})}{\partial U}\right] + \frac{C}{2m}(2U) \\
&= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}\frac{1}{h_\theta(x^{(i)})} + (1-y^{(i)})\frac{1}{1-h_\theta(x^{(i)})}\right)h_\theta(x^{(i)})(1-h_\theta(x^{(i)}))\frac{\partial(U^T a^{(i)} + b_2)}{\partial U}\right] + \frac{C}{m}U \\
&\quad \text{since } g'(x) = g(x)(1-g(x)) \\
&= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}\frac{1}{h_\theta(x^{(i)})} + (1-y^{(i)})\frac{1}{1-h_\theta(x^{(i)})}\right)h_\theta(x^{(i)})(1-h_\theta(x^{(i)}))\, a^{(i)}\right] + \frac{C}{m}U \\
&= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right)\, a^{(i)}\right] + \frac{C}{m}U \\
\frac{\partial J(\theta)}{\partial b_1} &= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}\frac{1}{h_\theta(x^{(i)})} + (1-y^{(i)})\frac{1}{1-h_\theta(x^{(i)})}\right)\frac{\partial h_\theta(x^{(i)})}{\partial U}\right] \\
&= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}\frac{1}{h_\theta(x^{(i)})} + (1-y^{(i)})\frac{1}{1-h_\theta(x^{(i)})}\right)h_\theta(x^{(i)})(1-h_\theta(x^{(i)}))\frac{\partial(U^T a^{(i)} + b_2)}{\partial U}\right] \\
&= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}\frac{1}{h_\theta(x^{(i)})} + (1-y^{(i)})\frac{1}{1-h_\theta(x^{(i)})}\right)h_\theta(x^{(i)})(1-h_\theta(x^{(i)}))\right] \\
\frac{\partial J(\theta)}{\partial W_{kj}} &= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right)\frac{\partial(U^T a^{(i)} + b_2)}{\partial W_{kj}}\right] + \frac{C}{2m}(2W_{kj}) \\
&= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right)U_k\frac{\partial f(z_k^{(i)})}{\partial W_{kj}}\right] + \frac{C}{m}W_{kj} \\
&= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right)U_k f'(z_k^{(i)})\frac{\partial(W_{k\cdot}x^{(i)} + (b_1)_k)}{\partial W_{kj}}\right] + \frac{C}{m}W_{kj} \\
&= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right)U_k f'(z_k^{(i)})x_j^{(i)}\right] + \frac{C}{m}W_{kj} \\
&= \frac{1}{m}\sum_{i=1}^{m}\left[\left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right)U_k(1-\tanh^2 z_k^{(i)})x_j^{(i)}\right] + \frac{C}{m}W_{kj}
\end{aligned}$$

[2] CS229 Lecture Notes 1 Page 16

$$
\begin{aligned}
\text{Let } \delta_k^{(i)} &= \left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right) U_k(1-\tanh^2 z_k^{(i)}) \\
\frac{\partial J(\theta)}{\partial W} &= \frac{1}{m}\sum_{i=1}^{m} \delta^{(i)} \left(x^{(i)}\right)^T + \frac{C}{m}W \\
\frac{\partial J(\theta)}{\partial (b_1)_i} &= \frac{1}{m}\sum_{i=1}^{m} \left[\left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right) \frac{\partial (U^T a^{(i)} + b_2)}{\partial (b_1)_i}\right] \\
&= \frac{1}{m}\sum_{i=1}^{m} \left[\left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right) U_k \frac{\partial f(z_k^{(i)})}{\partial (b_1)_i}\right] \\
&= \frac{1}{m}\sum_{i=1}^{m} \left[\left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right) U_k f'(z_k^{(i)}) \frac{\partial (W_{k\cdot} x^{(i)} + (b_1)_k)}{\partial W_{kj}}\right] \\
&= \frac{1}{m}\sum_{i=1}^{m} \left[\left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right) U_k f'(z_k^{(i)})\right] \\
\frac{\partial J(\theta)}{\partial b_i} &= \frac{1}{m}\sum_{i=1}^{m} \delta^{(i)} \\
\frac{\partial J(\theta)}{\partial L_k} &= \frac{1}{m}\sum_{i=1}^{m} \left[\left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right) \frac{\partial (U^T a^{(i)} + b_2)}{\partial L_k}\right] \\
&= \frac{1}{m}\sum_{i=1}^{m} \left[\left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right) \left(\sum_{j=1}^{H} U_j \frac{\partial f(z_j^{(i)})}{\partial L_k}\right)\right] \\
&= \frac{1}{m}\sum_{i=1}^{m} \left[\left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right) \left(\sum_{j=1}^{H} U_j f'(z_j^{(i)}) \frac{\partial (W_{j\cdot} x^{(i)} + (b_1)_j)}{\partial L_k}\right)\right] \\
&= \frac{1}{m}\sum_{i=1}^{m} \left[\left(-y^{(i)}(1-h_\theta(x^{(i)})) + (1-y^{(i)})h_\theta(x^{(i)})\right) \left(\sum_{j=1}^{H} U_j f'(z_j^{(i)}) W_{jk}\right)\right] \\
\frac{\partial J(\theta)}{\partial L} &= \frac{1}{m}\sum_{i=1}^{m} W^T \delta^{(i)}
\end{aligned}
$$