Notes for final report

Mitchell Whalen

March 25, 2024

Contents

1 dates

- schedule a brief (Zoom?) meeting to confirm "work distribution" and project details

1.1 report submission

- Monday, April 15th by the end of the day
- use LaTeX to generate the report
- Overleaf: useful way of writing a LaTeX doc in a group
- submit .tex, .pdf files and code files (in corresponding language)

1.2 oral component

- Wednesday, April 17th
- in person, sometime between 9:30 and 12:30
- just to chat about the report and ensure each group member participated as stated

2 part 1

- option B: handwritten digit recognition with neural networks

2.1 notes

- build and train two systems for digit classification: one neural network and one linear classifier
- having a validation set is optional but not required
- work with the MNIST dataset (should be available soon)
- the digits are already seperated into a training and test set
- recommended to divide the value of each by 255.0 to obtain data in the range [0, 1]
- some snippets of code will be provided to help (cite them properly, should you use them)

2.2 1-1 (5%)

- include 10 images of each of the digits
- matplotlib's subplot could be useful here

2.3 1-2 (10%)

- implement a function that computes the network shown in the diagram in the p1-optB.pdf doc
- the os should be linear combinations of the xs (i.e., the activation functions)

$$o_i = \sum_j w_{i,j} x_j + b_i \tag{1}$$

- include code in appendices

2.4 1-3 (10%)

- mini-batch gradient descent
- mini-batch gradient descent: unlikely to be seen in class; research it online
- cost function: sum of the negative log-probabilities of the correct answer for the N training cases under consideration (i.e., the negative log-likelihood of the training set)
- implement a function that computes the gradient of this cost function wrt. the parameters of the network W and b, for a given subset of the training cases
- you can (but don't have to) vectorize(?) your code
- include code in appendices

2.5 1-4 (5%)

- verify that you are computing the gradient in 1-3 correctly:
- compute the gradient using both your function as well as a finite-difference approximation

- do this for several coordinates(?) of the W and b
- include code and output in appendices, including both the precise gradient and the approximation

2.6 1-5 (15%)

- write code to minimize the cost function using mini-batch gradient descent, using the training set provided to you
- expect $>$91% correct classification; anything above 89% is acceptable
- suggested: learning rate of 0.01; batch size of 50
- for the training and test sets, graph the negative log-probability of the correct answer and correct classification rate versus the number of updates to the weights and biases during training, i.e., plot the learning curves
- display 20 digits which were classified correctly
- display 10 digits (from the test set) which were classified incorrectly

2.7 1-6 (10%)

- you can visualize what the network is doing by visualizing the Ws as if they were digits
- visualize each of the set of Ws that connect to o_0, the set of Ws that connect to o_1, etc. with each $w_{j,\,i}$ displayed at an appropriate pixel
- see image from p1-optB.pdf
- comment on the visualization of the weights that you obtained; tell us what you see (one or two sentences)

2.8 1-7 (25%)

- for this part a high-level library (such as Keras) is allowed, subject to approval
- implement a neural network with a hidden layer for digit classification
- implement the network sketched in p1-optB.pdf

- use tanh activation functions
- use 300 hidden units

2.9 1-8 (10%)

- use mini-batch gradient descent using the training set provided
- expect classification performance >95% correct classification
- suggested: learning rate 0.01; batch size of 50
- for the training and test sets, graph the correct classification rate versus the number of updates to the weights and biases during training, i.e., plot the learning curves
- display 20 digits which were classified correctly
- display 10 digits (from the test set) which were classified incorrectly

2.10 1-9 (10%)

- you can visualize what the network is doing by visualizing the Ws connected to the hidden layer as if they were digits
- select two interesting Ws (out of the total of 300) and explain what you think they are accomplishing (for each W)
- the explanations should be different for the different Ws

3 part 2

- option: error analysis implementation

3.1 notes

- determine which parts of the machine learning algorithm lead to test error
- perform error analysis akin to what appears in the error-analysis.pdf doc
- add an extra feature selector(?) to the pipeline from part 1 and "analyze its usefulness".

- or, do a different, challenging implementation assigned by the prof. Can be discussed in the meeting

3.2 error analysis

- plug in perfected data for each component, and see how accuracy changes
- this demonstrates which features really improve performance

3.3 ablative analysis

- explains difference between baseline (poorer) performance and current performance
- slowly remove features one by one to see how it affects performance

4 rubric

4.1 organization

- report is both accurate and compelling
- writing begins with an interesting or provocative introduction
- contains a clear and concise thesis statement
- body fully explores the topic and presents information in a sensible order
- conclusion restates the thesis or offers a comment or question on it
- effective and varied transitions link all ideas

4.2 elements of research

- writer provides facts and quotations from a variety of sources
- facts and ideas are either expressed in the writer's words, or else completely and correctly documented
- body of the report supports and develops the writer's thesis and it contains no extraneous ideas
- includes a complete and correct bibliography or source list

4.3 grammar, usage, mechanics and spelling

- few or no errors in mechanics, usage, grammar or spelling
- word choice is precise and appropriate for the audience