CS 510, Spring 2018, Second Half of Semester Review
May 2, 2018

The following is not an exhaustive list of what has been covered, and to be very explicit, the Final Exam may cover material discussed in class not mentioned below. That said, hopefully the following will be a helpful as you review the broad topics and techniques we have covered since the Midterm in CS510.

1. After the midterm, the first broad topic we covered was "bag of words" object recognition. While for most object recognition scenarios this approach is been supplanted by convolutional neural nets, it is still important historically and also for several of the key intellectual contributions that emerge. One of these contributions is a connection between document retrieval and object recognition. You are now in a strong position to explain in a few sentences how this connection was made and the corresponding major components of the approach as realized in documents and images.
2. In some respects, the presentation of how feature descriptors are used in the "bag of words" approach to object recognition presaged our use of features in programming assignment five. Clearly given the depth of practical experience you all acquired in programming assignment five using feature descriptors you could discuss the similarities and differences in your approach as compared to the "bag of words" approach.
3. In the “bag of words” approach the TF-IDF vector plays a critical role. It is therefore important that you understand how to compute this vector for a novel, i.e. probe, image.
4. The world of computer vision abounds with possible confusions in terms of approach. Here is one that you wish to stay very far away from: the "bag of words" approach to object recognition relies upon a fixed number of classes using a sequence of support vector machines. Be comfortable standing your ground and arguing that this is not generally a true characterization of the method.
5. There are two complementary ways to understand how a perceptron performs pattern recognition. One is of course the linear algebra underlying the algorithm, and the other is through pictures typically portraying different classes of data on a sheet of paper. The strength of having mastered both interpretations is the ability to relate one to the other. In particular, it's helpful in answering questions such as: "what is the unique solution found by a perceptron that separates two separable classes?"
6. When thinking about perceptrons is as good a time as any to reinforce your ability to explain in geometric terms how a support vector machine might accomplish a classification task solvable using a perceptron.
7. In the late 1960s and early 1970s there was a backlash in terms of enthusiasm for what a perceptron could and could not accomplish. Then, in the 1980s the pendulum swung back in favor of neural networks. As someone now familiar with the most basic aspects of neural networks as they relate to machine learning you're comfortable explaining the principal reasons for both of these historical events.
8. Assigning blame is a nontrivial problem in the context of multilayer neural networks. You should have no difficulty explaining this issue very precisely by comparing and contrasting a neural network node using a simple perceptron style threshold versus a sigmoid function.
9. In the second half of the semester we have used three distinct examples based in TensorFlow to both come to better understand the TensorFlow system itself and perhaps more important the behavior of neural nets employed to solve image recognition problems. The first of these examples illustrated explicitly the back propagation training procedure. This example is rich in detail and may serve you well as you go forward and need to remind yourself about precisely what takes place in this algorithm. so, to illustrate, you could explain in English the purpose of the following line of code: `d_z_2 = tf.multiply(diff, sigmaprime(z_2))`.
10. Thinking again about the TensorFlow backpropagation example, consider carefully your response to someone in a job interview asking you about the way in which back propagation finds the truly best neural network to solve problems such as the NIST hand written digit problem.
11. TensorFlow is first and foremost a dataflow language for performing tensor calculations. Two aspects of the statement require that you be fluent in your ability to explain to other people what you've learned. First, what is a tensor? Second, what is meant by a dataflow engine/machine.

12. It's important that you think hard about what it means to compute with a dataflow engine. As just one way of probing your understanding and that of others, answer this question: is it possible to actually construct a dataflow graph and yet compute nothing?
13. In the lecture discussing the use of neural nets to solve image recognition problems three reasons were given for networks from the 1980s (PDP-style) failing: too many weights, too few training samples, disregard for image geometry. You might have to explain this to someone else someday, be comfortable in your ability to elaborate on each of these points.
14. Repeatedly in the history of computer vision, the right data set with the right amount of ground truth comes along and lurches the entire field forward and in a new direction. The emergence of "ImageNet" beautifully illustrates this point. It's therefore very good thing that you can describe this data set and how it shifted the landscape of research in computer vision.
15. Before we ever actually used a convolutional neural network within TensorFlow we walked through an entire PowerPoint slide deck introducing the key elements of this contribution by Yaan LeCun. Highlights of this lecture included connections to human physiology, low-level features, scaling via image pyramids, and rectified linear units. These are all aspects of modern machine learning applied to image recognition that you can readily recognize and explain.
16. When it came to actually experimenting in TensorFlow with a convolutional neural network we drew upon a tutorial that used high-level abstractions such as Estimators and Layers. We spent considerable time with this code and indeed you have studied it further as part of the last programming assignment. Needless to say, you understand the purpose and basic functioning of essentially every line of code in this example.
17. Successfully building a learning model, essentially a complex data flow graph, to solve a recognition task is nontrivial. To make the process easier for us human beings, there is an entire second act in the overall TensorFlow story, namely TensorBoard. Being able to answer questions about what can and cannot be done in TensorBoard represents a basic understanding of how programmers use TensorFlow. Perhaps even more important, being able to motivate the existence of TensorBoard in terms of what it can teach about neural network learning is key.
18. The last major code example explored in depth during the semester implemented a neural network for again recognizing the NIST handwritten digits. What was most significant about this example was the deep connection to tensor board. As with the other two examples used this semester, it is an advantage to you that you now understand essentially every line of code in this example.
19. Recurrent neural networks are extremely useful for data that comes naturally as a series of recordings over time. Think for example about voice or video. Because we had a guest lecture on this topic, you have a basic understanding of how these networks differ from those used to assign labels to single images.
20. In spite of the incredible success being achieved using CNNs to solve computer vision problems, overfitting is a blight on the landscape. There several aspects of over fitting that you are now comfortable discussing. In very precise terms, it helps further motivate the understanding of the tri-partition approach to machine learning exemplified by: training, validation, testing. It also applies more generally to the pros and cons of K full cross validation and finally what might actually be expected when deploying a system into what amounts to a somewhat different environment.