INF5820/INF9820: Examples of exam questions

1 Spoken dialogue

1. Turn-taking is a crucial part of conversational competence. What linguistic and extra-linguistic factors can influence how people take and release turns, and where the boundaries of these turns are likely to lie?

2. Take the following utterance:

   robot please look at the ball no sorry the box

   Analyse the disfluent part based on Shriberg's disfluency model, and briefly describe how an NLU system could handle such disfluencies.

2 Speech recognition

Assume you have a user uttering the following utterance:
"Could you please take the red box and put it on the other end of the table?"

But that your speech recognition hypothesis turns out to be:
"Could you place vague red box and put it this on another end of that?"

Calculate the Word Error Rate (WER) between the ASR hypothesis and the actual utterance. Detail your calculations.

3 Natural language understanding

In our lectures, we have reviewed three distinct strategies for parsing spoken utterances. Describe these three strategies, and compare their respective advantages and shortcomings.

4 Dialogue management

Imagine a kitchen robot whose task is to ask the user what kinds of cereals he/she wants for breakfast, wait for the user answer, and then hand out the appropriate cereal box once it knows the desired cereal. We want to design a simple dialogue system to handle the interaction with the user. A simple way to model it is via a MDP with only two states: state $s = UnknownCereal$ where the robot doesn't know which cereal to give, and state $s = KnownCereal$, where the robot knows the cereal to hand out.

There are only two possible actions in the model:

- Action $a = AskCerealType$ corresponds to the robot asking the user for the cereal box he wishes to have. The action is only available in state $UnknownCereal$, and has a reward $R = -\mathbf{1}$ in that state.
- Action $a = GiveCereal$ corresponds to the robot physically giving the cereal to the user. The action is only available in state $KnownCereal$, and has a reward $R = +\mathbf{5}$ in that state.

When the robot executes the action $AskCerealType$ in state $UnknownCereal$, it has a probability **0.8** of reaching state $s = KnownCereal$ (if the user answers the robot's question), and a probability **0.2** of remaining in state $UnknownCereal$ (if the user ignores the question or provides an unclear answer). When the robot executes action $GiveCereal$ in state $KnownCereal$, the MDP reaches a final state and finishes.

You are asked to calculate the expected cumulative reward of asking the cereal type while in the $UnknownCereal$ state, i.e. $Q(s = UnknownCereal, a = AskCerealType)$. You can assume a discount factor of **0.9**.

(*Hint*: use Bellman equation to calculate the Q values).

5 Speech synthesis

Unit selection synthesis operates by searching for speech segments in a database that correspond to parts of the utterance to synthesise, and then gluing them together. Describe how this search is performed, and how the synthesiser ultimately decides which segments should be glued together.

6 Probabilistic modelling

You want to develop a spoken dialogue system for a human-robot interaction domain, where the user can tell the robot to move forward, backward, turn left and right, as well as take and release an object. You have already integrated a speech recogniser, but you quickly realise that it tends to make systematic mistakes for particular words. To reduce the number of speech recognition errors, you therefore decide to implement a simple post-processing tool to correct the N-Best lists provided by the speech recogniser. You decide to implement this post-processing tool using the *noisy-channel model*. This tool will take a N-Best list as input, and output another (hopefully improved) N-Best list.

As you remember from your dialogue system course, a noisy-channel model includes both a language model and a channel model. To estimate the language model, you collect a few sample sentences for your domain, and get the following tiny corpus of 20 sentences:

"Move forward"	"Turn right"	"Take object"	"Turn left"
"Release object"	"Move backward"	"Turn left"	"Take object"
"Move forward"	"Turn left"	"Take object"	"Move forward"
"Take object"	"Move forward"	"Release object"	"Move forward"
"Release object"	"Turn left"	"Turn right"	"Move forward"

You also need to find a channel model for your domain[1]. To this end, you analyse the outputs of the speech recogniser, and notice that most words have a probability 0.8 of being correctly recognised, and a probability 0.2 of being incorrectly recognised as another word. But there is an exception: the words "forward" and "backward" are frequently confused with one another by the ASR. These two words have a probability 0.5 of being correctly recognised, a probability 0.3 of being mistaken for the other one ("forward" instead of "backward" and vice versa), and a probability 0.2 of being yet another word.

Based on these informations, answer the following questions:

1. Derive a simple bigram model (without smoothing) for the small corpus shown above, and detail its probability distribution;

2. Construct the channel model corresponding to the word confusions informally described above, and detail its probability distribution;

3. Briefly explain how these two models are combined in a noisy channel model;

[1] To keep things simple, we consider here only 1-to-1 word confusions. A more general model would have to take into account more complex, n-to-m confusion matrices.

4. Finally, apply your post-processing tool to correct the following N-Best list:

$$\text{NBest list} = \left| \begin{array}{cc} \text{Move backward} & P = 0.6 \\ \text{Move forward} & P = 0.3 \\ \text{Turn right} & P = 0.1 \end{array} \right.$$

You can discard recognition hypotheses with a probability lower than 0.01.

7 Probabilistic modelling

Part 1

You want to build a new voice-controlled image repository for Android phones. Instead of using buttons to navigate through the images, the user will use voice controls to navigate through the pictures, using three distinct commands: "previous" (to move to the previous picture), "next" (to move to the next picture), and "delete" (to delete the current picture).

Instead of using a full-fletched speech recogniser, you decide to use a simpler system based on the detection of stop consonants at the beginning and end of each command, since these consonants nicely discriminate between the three commands ("previous" only has a stop at the beginning, "next" only at the end, and "delete" both at the beginning and the end).

We therefore have a variable $command$ with three distinct values = {*previous, next, delete*}, as well as two observation variables $stopDetectedAtBeginning$, $stopDetectedAtEnd$ with binary values. Of course, the relation between *command* and these two observation variables is probabilistic, so you want to estimate a probabilistic model between them. You start collecting data of users experimenting with your system, and you end up a sample of 1000 commands.

Out of these 1000 commands, 100 were "previous" commands, 200 were "delete" commands, and 700 were "next" commands. Furthermore:

- out of the 100 "previous" commands, 90 had a detected stop at the beginning, and 10 a detected stop at the end;
- out of the 200 "delete" commands, 180 had a detected stop at the beginning, and 180 had a detected stop at the end;
- out of the 700 "next" commands, 50 had a detected stop at the beginning, and 600 had a detected stop at the end.

Given this information:

1. construct a Bayesian Network representing the three random variables and their probability distributions;

2. Based on this Bayesian Network, calculate the probability that the user uttered "delete" if the system detected both a stop at the beginning and at the end – that is, $P(command = delete | stopDetectedAtBeginning = true, stopDetectedAtEnd = true)$.

Part 2

The Bayesian Network you constructed allows us to determine the probability of each user command given the observations of stop consonants. We haven't however determined yet how our application will make a *decision* about its actions based on it. The system has four actions at its disposal: *systemAction* = {*GoToPrevious, GoToNext, Delete, DoNothing*}. Each of these actions have different utilities – for instance, deleting a picture if the user intended something else should have a high negative utility.

Assume the following utility distribution:

command	*systemAction*	Utility
previous	GoToPrevious	+1
previous	GoToNext	-2
previous	Delete	-5
next	GoToPrevious	-2
next	GoToNext	+1
next	Delete	-5
delete	GoToPrevious	-2
delete	GoToNext	-2
delete	Delete	+2

The *DoNothing* action always has a utility of 0.0.
Given this utility function, answer the following questions:

3. Assumes the system detects a stop consonant at the beginning but no stop at the end (i.e. $stopDetectedAtBeginning = true, stopDetectedAtEnd = false$). What would then be the best action for the system to select, based on the utility distribution shown above? Show your calculations.

4. Draw the Bayesian network augmented with utility nodes (diamonds) and decision nodes (squares) that represents the full problem.