MODEL SOLUTIONS

SETTER: Lucia Specia / Mark Hepple

Data Provided: None

DEPARTMENT OF COMPUTER SCIENCE **September/October 2014**

TEXT PROCESSING **2 hours and 30 minutes**

Answer the question in Section A, and THREE questions from Section B.

All questions carry equal weight. Figures in square brackets indicate the percentage of available marks allocated to each part of a question.

SECTION A

1. a) Describe and explain the TF.IDF term weighting scheme used in information retrieval. Include the formula (or formulae) for computing TF.IDF values as part of your answer. [30%]

ANSWER:

[P1] F.IDF assigns weights to terms by taking into account two elements: the frequency of that term in a particular document and the proportion of documents in the corpus in which it occurs. The TF part prefers terms which occur frequently in a document. The IDF part gives extra weight to terms which do not occur in many documents in the collection, based on the intuition that terms that appear very widely in the collection will be a poor discriminator of the documents most relevant to a user's information need. [P2] The idf value for a term w is computed as follows, where D is the set of documents in the document collection and $df_{w,D}$ is the document frequency of w (the count of documents in which w appears):

$$idf_{w,D} = log\frac{|D|}{df_{w,D}}$$

The TF.IDF value for a given term (w) in a given document (d) is the product of the term's frequency in d ($tf_{w,d}$) and its IDF value, i.e.:

$$tf.idf_{w,d,D} = tf_{w,d} \cdot idf_{w,D}$$

b) Compression techniques are important due to the growth in volume of the data that must be stored and transmitted.

(i) Explain the difference between **lossy** and **lossless** forms of compression. Discuss the suitability of these alternative forms of compression for different media types (e.g. for text vs. image data). [10%]

ANSWER:

Lossless data compression refers to the class of data compression algorithms that allow the original data to be perfectly reconstructed from the compressed data. By contrast, lossy data compression methods achieve data reduction by discarding (losing) information.
For an example of the latter, we can reduce the data volume of an image that has a high pixel-density by reducing the pixel density. The image that results is typically still interpretable as an image, even if it is of lower quality/fidelity. Such approaches are commonly applied to image, video and audio data (especially for use in streaming contexts). For such media, a substantial amount of data can be discarded before the result is sufficiently degraded for this to be noticed by the user.
Text compression applications require *lossless* compression methods — the idea of a version of text from which N% of the information has been discarded doesn't make sense. In general, we expect decompression to return a text that is identical to the original in both content and form.

(ii) Explain the difference between **static**, **semi-static** and **adaptive** techniques for text compression, noting their key advantages and disadvantages. [10%]

ANSWER:

Compression techniques can also be distinguished by whether they are

- **Static** – use a fixed model or fixed dictionary derived in advance of any text to be compressed
  - adv: model does not need to be transmitted
  - disadv: model may not be well suited to text currently being compressed
- **Semi-static** – use current text to build a model or dictionary during one pass, then apply it in second pass
  - adv: model should be well suited to current text
  - disadv: model must also be transmitted, reducing effectiveness of compression

- **Adaptive** – build model or dictionary adaptively during one pass
  - adv: model does not need to be transmitted
  - disadv: decoder determines model used at each stage from data decoded so far, so cannot do random access into data

c) What is the noisy channel model? Give a diagram of the model as part of your answer. Suggest a text processing context where the noisy channel model has been used. [30%]

ANSWER:

[P1] Using information theory, Shannon modeled the goal of communicating down a telephone line – or in general across any channel – in the following way. Imagine communication along a low-quality telephone line where the sender creates a message which is being transmitted over a channel with limited capacity; the receiver has to guess what the original message was. It is assumed that the output of the channel depends probabilistically on the input. The goal is to encode the message in such a way that it occupies minimal space while still containing enough redundancy to be able to detect and correct errors. On receipt, the message is decoded to give what was most likely the original message. [P2]

$$\xrightarrow{W} \boxed{\text{Encoder}} \xrightarrow{X} \boxed{\begin{array}{c}\text{Noisy Channel}\\ p(y|x)\end{array}} \xrightarrow{Y} \boxed{\text{Decoder}} \xrightarrow{\hat{W}}$$

[P3] A version of the noisy channel model has been used to model the machine translation process. As an example, suppose we want to translate a text from English to French. The channel model for translation assumes that the true text is French, but that unfortunately when it was transmitted to us, it went through a noisy communication channel and came out as English. All we need to do in order to translate is to recover the original French, i.e., to decode the English to get the French.

d) Explain the **two** most common (semi-)automated approaches to expand sets of seed opinion words (like “good” and “bad”) with more opinion words to create lexica for Sentiment Analysis: dictionary-based and corpus-based approaches. Give **one** advantage and **one** disadvantage of each approach. [20%]

ANSWER:

[P1] The two most common approaches to create lexica for Sentiment Analysis are 1) lookup in dictionaries or lexical databases like WordNet for synonyms in both sets of seed words to expand those sets, for antonyms in the negative set (generating positive words) and in the positive set (generating negative words). This can be repeated once the sets become larger. 2) corpus-based approaches where patterns are built from the seed words, such as “good and ”, “fragile but ”, “very tasty and ”, and searched on corpora (such as the Web). In this case, all examples of patterns should result in more positive words or phrases (such as “good and reliable”). [P2] The advantages of the first approach are (only one is necessary) simplicity, as searches are straightforward, and accuracy, as lexical databases are normally built by humans and tend to be accurate. The disadvantages (only one is necessary) are that lexical databases are only available for a limited number of languages, and that they usually contain only words, not phrases. The advantages (only one is necessary) of the corpus-based approach is that it is more flexible (patterns can include phrases), and that corpora of this type are available for most languages. The disadvantages (only one is necessary) are that the results are not always accurate, and that patterns must be cleverly elaborated to avoid noisy results.

SECTION B

2. a) Describe *manual* and *automatic* approaches to indexing for Information Retrieval. Discuss the advantages and disadvantages of these two approach? [20%]

ANSWER:

[P1] In manual approaches documents are indexed using terms that are identified manually, often from some pre-defined controlled vocabulary or taxonomy. In automatic approaches the index terms are identified directly from the documents.

[P2] One advantage and one disadvantage are enough: The advantage of manual indexing is that the index terms that are added are more closely controlled than is possible with automatic indexing. They are more likely to be appropriate for the document and unambiguous. The (human) indexer may also be able to use judgement and experience to identify index terms which are appropriate but could not be easily identified by simple analysis of the document. On the other hand manual indexing is expensive in terms of effort that is required from human indexers which can make this approach impractical. Manual indexing also allows the use of hierarchical indexing terms which can be used to create complex queries and this is difficult to achieve automatically. However, the index terms may be drawn from some controlled vocabulary which a user may not be familiar with and training may be required to search a manually indexed collection effectively.

The advantages and disadvantages of automatic approaches are largely the opposite of manual indexing. Automatic indexing does not require manual effort and can be efficiently applied to large document collections but, on the other hand, the quality of the index terms is unlikely to be as good as those that are added manually.

In the context of Information Retrieval, given the following two documents:

Document 1: Sea shell, buy my sea shell!

Document 2: You can buy lovely SEA SHELL at the sea produce market.

and the query:

Query 1: buy lovely sea shell

b) Explain three types of manipulations (except term weighting) that can be done on document terms before indexing them. What are the advantages and disadvantages (if any) of each of these manipulations? [20%]

ANSWER:

Term manipulations can include any of the following (answer only need 3): capitalisation, stemming, stop-word removal, indexing multi-terms, normalisation. Three exemplar definitions:

A *stoplist* is a list of words ('stop-words') that are ignored when documents are indexed, and likewise discounted from queries. These are words that are so widespread in the document collection that they are of v.little use for discriminating between documents that are/are not relevant to a query. Their exclusion eliminates a large number of term occurrences that would need to be recorded, thereby reducing the size of indexes and saving computational effort during both indexing and retrieval.

Stemming refers to the process of reducing words that are morphological variants to their common root or stem, e.g. so that variants *computer, computes, computed, computing*, etc. are reduced to a stem such as *compute*. For IR, stemming is applied to documents before indexing and to queries. The effect of this is that when a query contains a term such as *computing*, retrieval can potentially return documents on the basis of their containing any of the morphological variants of the same root. This is the intended key benefit of using stemming, although its usefulness is debated. Stemming will also produce some reduction in the size of document indexes.

Capitalisation refers to the process of normalising the case of words so that a single case is used, for example, all words are lowercased (e.g. SHELL = shell). This procedure makes indexing and retrieval more efficient by decreasing the number of terms that have to be represented. It also makes the term weighting more reliable, as higher frequency counts will be observed when putting together different variants of the term. One possible disadvantage is that intended capitalisation in terms for disambiguation purposes will be disregarded, e.g. Turkey (country) vs turkey (bird).

c) Applying stop word removal and capitalisation, show how Document 1 and Document 2 would be represented using an *inverted index*. Provide the stoplist used. [10%]

ANSWER:

Based on *lowercasing* and *stopword removal* assuming a stoplist that includes the following terms: {`at, can, my, the, you, !, ,, .`}, we get the following inverted index:

term-id	word	docs
1	buy	d1:1, d2:1
2	lovely	d2:1
3	market	d2:1
4	produce	d2:1
5	sea	d1:2, d2:2
6	shell	d1:2, d2:1

d) Assuming *term frequency* (TF) is used to weight terms, compute the similarity between each of the two documents (Document 1 and Document 2) and Query 1. Compute this similarity using two metrics: Cosine and Euclidean. Determine the ranking of the two documents according to each of these metrics and discuss any differences in the results. [25%]

ANSWER:

Using the term order taken from the inverted index above, we can represent the two documents and the query as vectors as follows:

d1: $\langle\, 1, 0, 0, 0, 2, 2 \,\rangle$
d2: $\langle\, 1, 1, 1, 1, 2, 1 \,\rangle$
q: $\langle\, 1, 1, 0, 0, 1, 1 \,\rangle$

Ranking using cosine similarity: The cosine between two vectors $\vec{x}$ and $\vec{y}$ is computed as:

$$cos(\vec{x}, \vec{y}\,) = \frac{\vec{x} \cdot \vec{y}}{|\vec{x}||\vec{y}|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}$$

The vector magnitudes and cosine values are then:

$$|\mathrm{d1}| = \sqrt{1^2 + 0^2 + 0^2 + 0^2 + 2^2 + 2^2} = \sqrt{9} = 3$$

$$|\mathrm{d2}| = \sqrt{1^2 + 1^2 + 1^2 + 1^2 + 2^2 + 1^2} = \sqrt{9} = 3$$

$$|\mathrm{q}| = \sqrt{1^2 + 1^2 + 0^2 + 0^2 + 1^2 + 1^2} = \sqrt{4} = 2$$

$$cos(\mathrm{d1}, \mathrm{q}) = \frac{1.1 + 0.1 + 0.0 + 0.0 + 2.1 + 2.1}{3.2} = 5/6$$

$$cos(\mathrm{d2}, \mathrm{q}) = \frac{1.1 + 1.1 + 1.0 + 1.0 + 2.1 + 1.1}{3.2} = 5/6$$

Thus, the cosine values computed for the two documents are the same, and so the two are equally rated as relevant to the query.

Ranking using euclidean distance: Euclidean distance is computed as $\sqrt{\sum_{i=1}^{n}(q_i - d_i)^2}$. In this case:

$$dis(\mathrm{d1}, \mathrm{q}) = \sqrt{(1-1)^2 + (0-1)^2 + (0-0)^2 + (0-0)^2 + (2-1)^2 + (2-1)^2} = \sqrt{3}$$

$$dis(\text{d2}, \text{q}) = \sqrt{(1-1)^2 + (1-1)^2 + (1-0)^2 + (1-0)^2 + (2-1)^2 + (1-1)^2} = \sqrt{3}$$

The difference between the two metrics: while in this example the two metrics behave in the same way and they are not able to distinguish the two documents, in general cosine is a better metric for IR because it normalises the differences between the query and document vectors by the length of such vectors. This is important because documents can vary in size and a document should not be considered more relevant simply because it is closer in length to the query. Essentially the cosine metric computes the angle between two vectors, and therefore the length of such vectors is less relevant.

e) Discuss the expected effect of using TF.IDF to weight the terms in Document 1 and Document 2: would this be a better term weighting scheme in this example? Include the formula (or formulae) for computing TF.IDF values as part of your answer.

[25%]

ANSWER:

TF.IDF assigns weights to terms by taking into account two elements: the frequency of that term in a particular document (TF) and the proportion of documents in the corpus in which it occurs. The IDF for a term w is computed as follows, where D is the set of documents in the document collection and df_w is the document frequency of w (the count of documents in which w appears):

$$idf_{w,D} = log\frac{|D|}{df_w}$$

The TF.IDF value for a given term (w) in a given document (d) is:

$$tf.idf_{w,d,D} = tf_{w,d} \cdot idf_{w,D}$$

In this example, using TF.IDF would set the weight of three of the query terms to 0: *buy, sea, shell*, since they appear in all (two) documents in the collection and thus $log\frac{|D|}{df_w} = log\frac{2}{2} = log(1) = 0$. The only query term that receives a weight different from 0 is *lovely*, which only happens in Document 1, and therefore this document is ranked first. TF.IDF has a positive effect in this example and in general, since it disregards terms that are frequent across the whole collection of indexed documents.

3. a) Explain the three main approaches to Machine Translation: *direct*, *transfer* and *interlingual*. [20%]

ANSWER:

The key difference between the three approaches is the level of analysis which is applied to the source text.

Direct approaches apply very little analysis to the the source text and rely on simple translation of each word in the source text. Statistical MT could be considered to be a direct approach.

Transfer approaches attempt to analyse the structure of the source text to produce an intermediate representation. The intermediate representation of a sentence from the input text is then used in generating the translated sentence in the target language. Transfer approach can employ syntactic and/or semantic representations.

Interlingual approaches rely on a representation of the meaning which is independent of both the source and target language. The source sentence goes through syntactic and semantic analysis to be translated into the interlingua. This representation is then used to generate a sentence in the target language. The difference between transfer approaches which use semantic representations and interlingua approaches rests on the independence of the system used to represent meaning; interlinguas are completely independent of the source and target languages while the representation used in semantic transfer simply aims to capture enough information to allow translation.

b) When applied to translating from French to English, the statistical paradigm to statistical machine translation might be expressed by the following equation:

$$E^* \quad = \quad \underset{E}{\operatorname{argmax}}\ P(E) \cdot P(F|E)$$

Explain what this equation means, and indicate the role played by the components $P(E)$ and $P(F|E)$ in the process of translation. [20%]

ANSWER:

[P1] The equation tells us that the optimal English translation E^* of a French sentence F is found by identifying the English sentence E that maximises the equation, which has two components, that serve different purposes, as follows. [P2] The (monolingual) Language Model (LM) $P(E)$ favours candidate Es that are good strings of English, i.e. this component provides for the *fluency* of translations. The Translation Model (TM) $P(F|E)$ tests if candidate Es are likely translations of F, i.e. it provides for *faithfulness* in translation.

c) Describe the following metrics for evaluating Machine Translation systems: BLEU and round-trip translation. Discuss the advantages and disadvantages of automatic evaluation metrics like these two over manual evaluation metrics. [20%]

ANSWER:

[P1] BLEU: this metric compares n-grams in the system output to n-grams in one or more human translations. N-grams normally go from 1 to 4 words. The higher the number of overlapping n-grams, the closer the system output is to a human output, and therefore, the better the translation is considered. n-grams are weighed proportionally to their length: matches of longer n-grams count more towards the final score.

The round-trip translation or back-translation evaluation approach translates the original text, written in L1, into another language L2 and then back into L1. The quality of the translation is evaluated by checking how close the text produced is to the original, source text.

[P2] Automatic metrics are much cheaper: once a gold-standard set is collected, they can be repeated for any MT system, they are fast (a matter of seconds to score a dataset), and they are deterministic: different rounds on the same text will always produce the same figures, which is not the case with human judges (disagreements exist both between different judges and the same judge at different points in time).

d) Give an example of a language pair for which Statistical Machine Translation is likely to work well and another example for which is it likely to work badly. Explain your choices with reference to the approach used by Statistical Machine Translation systems. [20%]

ANSWER:

SMT is best suited to languages which are structurally similar (e.g. English-French, Spanish-Portuguese) and least suited to those which are structurally different (e.g. English-Chinese). The reason is that it is difficult to align parallel text when the languages include pairs which are structurally different and accurate alignment is important for the translation model used by SMT. Reasons structural differences can make alignment difficult include (1) different notions of what a words is (so the alignment between words is not one to one), (2) different word orders (leading to a huge number of possible alignments which would be impossible to compute), (3) different morphologies (making it difficult to learn translation probabilities).

e) Explain the difference between Hierarchical Phrase-based Machine Translation models and standard Phrase-based Statistical Machine Translation models. Do Hierarchical Phrase-based Machine Translation models use linguistic information, and if so, of what type? [20%]

ANSWER:

[P1] Hierarchical SMT allows to introduce structure into phrase-based SMT models. Instead of representing only flat phrases with corresponding source and target words/phrases, these models allow the representation, in the phrase table, of phrases with variables which, at decoding time, can be substituted by other phrases/words in the phrase table, until no more variables exist.

[P2] These models do not use linguistic information. Rules often use a single variable, X, only to cover for hierarchical structures.

4. a) Given sentences like the following:

- *My new phone works well and it is much faster than the old one.*
- *My new phone has 32GB of memory and can play videos.*

What is the first step to detect the sentiment in these two sentences? Should both these sentences be addressed in the same way by sentiment analysis approaches? [10%]

ANSWER:

[P1] The first step is to run a subjectivity analysis. This has to do with whether the text (word/phrase, sentence, document) contains opinions, emotions, sentiment, or simply facts. Only subjective sentences should then be put forward for sentiment analysis, which has to do with the actual polarity of the text (word/phrase, sentence, document): positive, negative, or more fine-grained distinctions. [P2] In the example, only the first sentence is subjective and therefore has a sentiment that can be analysed.

b) Explain a common approach for subjectivity analysis. [10%]

ANSWER:

A simple rule-based subjectivity classifier can be built: a sentence/document is subjective if it has at least n (say 2) words that belong to an emotion words lexicon; a sentence/document is objective otherwise.

c) Discuss the relevance of automatic techniques for sentiment analysis for marketing purposes. [10%]

ANSWER:

Extracting opinions, sentiments and emotions expressed by humans in texts and using this information for marketing purposes allow companies to process large volumes of textual data, such as customer reviews, and identify products that are more likely to be successful with certain audiences, or that need more marketing or even that should be taken out of the market. This can be directly used to place ads in the user-generated content, for example when the user praises a product. Processing these huge amounts of data manually is not possible, and the old style of gathering this data (interviews, etc.) reaches far smaller audiences.

d) Given the following sentences and opinion lexicon (adjectives only), apply the weighted lexical-based approach to classify EACH sentence as **positive**, **negative** or **objective**. Show the final emotion score for each sentence, but also how it was generated.

In addition to use of the lexicon, make sure you consider any general rules that have an impact in the final decision. Explain these rules when they are applied. [20%]

Lexicon:

awesome	5
boring	-3
brilliant	2
funny	3
happy	4
horrible	-5

(S1) He is brilliant and funny.
(S2) I am not happy with this outcome.
(S3) I am feeling AWESOME today, despite the horrible comments from my supervisor.
(S4) He is extremely brilliant but boring, boring, very boring.

ANSWER:

The general weighted lexical-based approach counts positive (Cpos) and negative (Cneg) words in the text and weights them using the weights in the lexicon given:
If Cpos $>$ Cneg then positive
If Cpos $<$ Cneg then negative
If Cpos $=$ Cneg $=$ 1 then objective

(S1) Emotion(brilliant) = 2; Emotion(funny) = 3. Therefore Cpos = 2+3 and Cneg = 0, so Cpos > Cneg = positive

(S2) Emotion(happy)= 4; "not" is detected in neighbourhood (of 5 words around). Emotional valence of term is decreased by 1 and sign is inverted. Therefore Emotion(happy)=-3, and Cneg=-3, so Cneg > Cpos = negative

(S3) Emotion(horrible) = -5, Emotion(awesome) = 5, but "awesome" is intensified because it is in capital letters, and in this case it's intensified by 1 (because it's a positive word). Therefore, Cneg = -5, Cpos = 6, so Cpos > Cneg = positive

(S4) Emotion(brilliant) = 2; Emotion(boring) = -3, but it happens 3x, so Emotion(boring) = -9. "extremely" is a (positive) intensifier in this case, with +2 added, so Emotion(brilliant) = 4. Cpos = 4 and Cneg = -9, so Cneg > Cpos = negative

e) According to Bing Liu's model, an **opinion** is said to be a quintuple $(o_j, f_{jk}, so_{ijkl}, h_i, t_l)$. Explain each of these elements and exemplify them with respect to the following text. Identify the features present in the text, and for each indicate its sentiment value as either *positive* or *negative*. Discuss two language processing challenges in automating the identification of such elements. [30%]

> “I have just bought the new iPhone 5. It is a bit heavier than the iPhone 4, but it is much faster. The camera lenses are also much better, taking higher resolution pictures. The only big disadvantage is the cost: it is the most expensive phone in the market. Lucia Specia, 12/08/2014.”

ANSWER:

[P1]

- o_j is a target object: iPhone 5
- f_{jk} is a feature of the object o_j: weight, speed, camera/lenses/pictures, price
- so_{ijkl} is the sentiment value of the opinion: negative, positive, positive, negative
- of the opinion holder h_i (usually the author of the post): Lucia Specia
- on feature f_{jk} of object o_j at time t_l: 12/08/2014.

[P2] Some of the challenges are (only two necessary):

- Need Named Entity Recognition to identify target objects.
- Need Information Extraction to extract features of the target object (properties of objects), as well as time and holder.
- Need co-reference resolution to know that "it“ = iPhone.
- Need synonym match when words used do not belong to lexicon of opinion words, e.g. fast versus efficient

f) Explain three metrics to evaluate the quality of binary (negative/positive) sentiment analysis systems. Give their intuitions and show their formulae. [20%]

ANSWER:

Based on an a gold-standard dataset (aka test corpora): i.e., text segments that have been classified by humans into positive vs negative cases, we can compare the system output to human classifications and compute metrics like:

Accuracy = number of correctly classified segments / number of segments, where correctly classified segments are those where the system agrees with the human decision (i.e., both positive or both negative)

Precision Positive = number of segments correctly classified as positive / number of segments classified as positive (idem for Precision Negative)

Recall Positive = number of segments correctly classified as positive / number of positive segments (idem for Recall Negative)

F-measure Positive = (2 * Precision Positive * Recall Positive) / Precision Positive + Recall Positive (idem for F-measure Negative)

5. a) Text compression techniques are important because growth in volume of text continually threatens to outstrip increases in storage, bandwidth and processing capacity. Briefly explain the differences between:

(i) **symbolwise** (or statistical) and **dictionary** text compression methods; [10%]

ANSWER:

- **Symbolwise methods** work by estimating the probabilities of symbols (characters or words/non-words) and coding one symbol at a time using shorter codewords for the more likely symbols
- **Dictionary methods** work by replacing word/text fragments with an index to an entry in a dictionary

(ii) **modelling** versus **coding** steps; [10%]

ANSWER:

Symbolwise methods rely on a modeling step and a coding step

- **Modeling** is the estimation of probabilities for the symbols in the text – the better the probability estimates, the higher the compression that can be achieved
- **Coding** is the conversion of the probabilities obtained from a model into a bitstream

b) The script for the fictitious language Gavagese contains only the 7 characters a, e, u, k, r, f, d. You assemble a large electronic corpus of Gavagese and now want to compress it. You analyse the frequency of occurrence of each of these characters in the corpus and, using these frequencies as estimates of the probability of occurrence of the characters in the language as a whole, produce the following table:

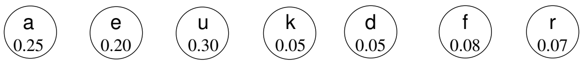
Symbol	Probability
a	0.25
e	0.20
u	0.30
k	0.05
r	0.07
f	0.08
d	0.05

(i) Show how to construct a Huffman code tree for Gavagese, given the above probabilities. [30%]

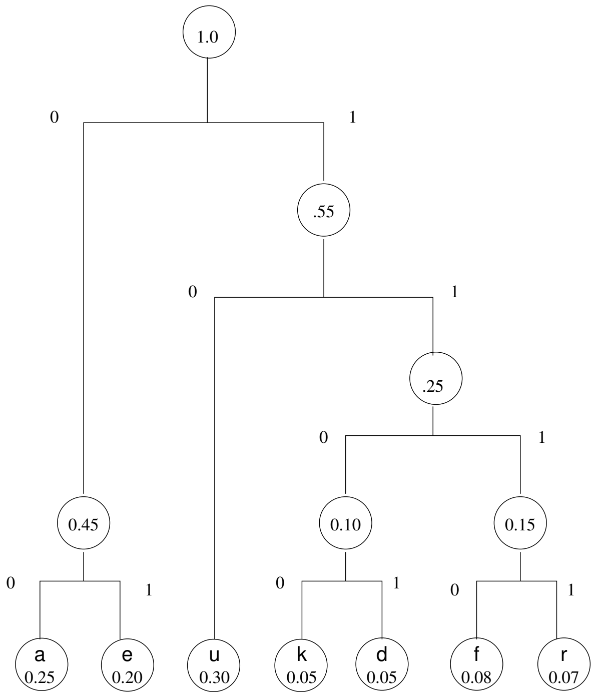
ANSWER:

Start off by creating a leaf node for each character, with associated probability (a). Then join two nodes with smallest probabilities under a single parent node, whose probability is their sum, and repeat till only one node left. Finally, 0's and 1's are assigned to each binary split.

(a)



(b)



(ii) Use your code tree to encode the string *dukerafua* and show the resulting binary encoding. For this string, how much length does your codetree encoding save over a minimal fixed length binary character encoding for a 7 character alphabet? [10%]

ANSWER:

Encoding for *dukerafua* will be

```
d    u  k    e  r    a  f    u  a
1101 10 1100 01 1111 00 1110 10 00
```

For a seven letter alphabet a minimal fixed length binary character encoding is 3 bits per character. There are 9 characters in the string, so a fixed length encoding would require 27 characters. The codetree encoding is 26, so one character only is saved (the advantages will become apparent over larger more statistically representative strings).

c) One popular compression technique is the LZ77 method, used in common compression utilities such as *gzip*.

(i) Explain how LZ77 works. [25%]

ANSWER:

The **key idea** underlying the LZ77 adaptive dictionary compression method is to replace substrings with a pointer to previous occurrences of the same substrings in same text. The encoder output is a series of triples where

- the first component indicates how far back in decoded output to look for next phrase
- the second indicates the length of that phrase
- the third is next character from input (only necessary when not found in previous text, but included for simplicity)

Issues to be addressed in implementing an adaptive dictionary method such as LZ77 include

- how far back in the text to allow pointers to refer
  - references further back increase chance of longer matching strings, but also increase bits required to store pointer
  - typical value is a few thousand characters
- how large the strings referred to can be
  - the larger the string, the larger the width parameter specifying it
  - typical value $\sim$ 16 characters
- during encoding, how to search window of prior text for longest match with the upcoming phrase
  - linear search very inefficient
  - best to index prior text with a suitable data structure, such as a trie, hash, or binary search tree

A popular high performance implementation of LZ77 is **gzip**

- uses a hash table to locate previous occurrences of strings
  - hash accessed by next 3 characters
  - holds pointers to prior locations of the 3 characters
- pointers and phrase lengths are stored using variable length Huffman codes, computed semi-statically by processing 64K blocks of data at a time
- pointer triples are reduced to pairs, by eliminating 3rd element
  - first transmit phrase length – if 1 treat pointer as raw character; else treat pointer as genuine pointer

(ii) How would the following LZ77 encoder output

$$\langle 0,0,b\rangle\langle 0,0,a\rangle\langle 0,0,d\rangle\langle 3,3,b\rangle\langle 1,3,a\rangle\langle 1,3,d\rangle\langle 1,3,a\rangle\langle 11,2,a\rangle$$

be decoded, assuming the encoding representation presented in the lectures? Show how your answer is derived. [15%]

ANSWER:

1. $\langle 0,0,b\rangle$ Go back 0 copy for length 0 and end with b: b
2. $\langle 0,0,a\rangle$ Go back 0 copy for length 0 and end with a: ba
3. $\langle 0,0,d\rangle$ Go back 0 copy for length 0 and end with a: bad
4. $\langle 3,3,b\rangle$ Go back 3 copy for length 3 and end with b: $badbadb$
5. $\langle 1,3,a\rangle$ Go back 1 copy for length 3 and end with a: $badbadbbbba$
6. $\langle 1,3,d\rangle$ Go back 1 copy for length 3 and end with d: $badbadbbbbaaaad$
7. $\langle 1,3,a\rangle$ Go back 1 copy for length 3 and end with a: $badbadbbbbaaaadddda$
8. $\langle 11,2,a\rangle$ Go back 11 copy for length 2 and end with a: $badbadbbbbaaaaddddabba$

Thus, the decoded string is:

`badbadbbbbaaaaddddabba`

END OF QUESTION PAPER