Technical Report

UCAM-CL-TR-790
ISSN 1476-2986

Number 790

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Automated assessment of ESOL free text examinations

Ted Briscoe, Ben Medlock, Øistein Andersen

November 2010

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

Technical reports published by the University of Cambridge
Computer Laboratory are freely available via the Internet:

http://www.cl.cam.ac.uk/techreports/

ISSN 1476-2986

Automated Assessment of ESOL Free Text Examinations

Ted Briscoe
Computer Laboratory
University of Cambridge
and
Ben Medlock and Øistein Andersen
iLexIR Ltd
ejb@cl.cam.ac.uk ben/oistein@ilexir.co.uk

Abstract

In this report, we consider the task of automated assessment of English as a Second Language (ESOL) examination scripts written in response to prompts eliciting free text answers. We review and critically evaluate previous work on automated assessment for essays, especially when applied to ESOL text. We formally define the task as discriminative preference ranking and develop a new system trained and tested on a corpus of manually-graded scripts. We show experimentally that our best performing system is very close to the upper bound for the task, as defined by the agreement between human examiners on the same corpus. Finally we argue that our approach, unlike extant solutions, is relatively prompt-insensitive and resistant to subversion, even when its operating principles are in the public domain. These properties make our approach significantly more viable for high-stakes assessment.

1 Introduction

The task of automated assessment of free text passages or essays is distinct from that of scoring short text or multiple choice answers to a series of very specific prompts. Nevertheless, since Page (1966) described the Project Essay Grade (PEG) program, this has been an active and fruitful area of research. Today there are at least 12 programs and associated products (Williamson, 2009), such as the Educational Testing Service's (ETS) e-Rater (Attali and Burstein, 2006), PearsonKT's KAT Engine / Intelligent Essay Assessor (IEA) (Landauer *et al*, 2003) or Vantage Learning's Intellimetric (Elliot, 2003), which are deployed to assess essays as part of self-tutoring systems or as a component of examination marking (e.g. Kukich, 2000). Because of the broad potential application of automated assessment to essays, these systems focus as much on assessing the semantic relevance or 'topicality' of essays to a given prompt as on assessing the quality of the essay itself.

Many English as a Second Language (ESOL) examinations include free text essay-style answer components designed to evaluate candidates' ability to write, with a focus on specific communicative goals. For example, a prompt might specify writing a letter to a friend describing a recent activity or writing an email to a prospective employer justifying a job application. The design, delivery, and marking of such examinations is the focus of considerable research into task validity for the specific skills and levels of attainment expected for a given qualification (e.g. Hawkey, 2009). The marking schemes for such writing tasks typically emphasise use of varied and effective language appropriate for the genre, exhibiting a range and complexity consonant with the level of attainment required by the examination (e.g. Shaw and Weir, 2007). Thus, the marking criteria are not primarily prompt or topic specific but linguistic. This makes automated assessment for ESOL text (hereafter AAET) a distinct subcase of the general problem of marking essays, which, we argue, in turn requires a distinct technical approach, if optimal performance and effectiveness are to be achieved.

Nevertheless, extant general purpose systems, such as e-Rater and IEA have been deployed in self-assessment or second marking roles for AAET. Furthermore, Edexcel, a division of Pearson, has recently announced that from autumn 2009 a revised version of its Pearson Test of English Academic (PTE Academic), a test aimed at ESOL speakers seeking entry to English speaking universities, will be entirely assessed using "Pearson's proven automated scoring technologies"[1]. This announcement from one of the major providers of such high stakes tests makes investigation of the viability and accuracy of automated assessment systems a research priority[2]. In this report, we describe research undertaken in collaboration with Cambridge ESOL, a division of Cambridge Assessment, which is, in turn, a division of the University of Cambridge, to develop an accurate and viable approach to AAET and to assess the appropriateness of more general automated assessment techniques for this task.

Section 2 provides some technical details of extant systems and considers their likely efficacy for AAET. Section 3 describes and motivates the new model that we have developed for AAET based on the paradigm of discriminative preference ranking using machine learning over linguistically-motivated text features automatically extracted from scripts. Section 4 describes an experiment training and testing this classifier on samples of manually marked scripts from candidates for Cambridge ESOL's First Certificate of English (FCE) examination and then comparing performance to human examiners and to our reimplementation of the key component of PearsonKT's IEA. Section 5 discusses the implications of these experiments within the wider context of operational deployment of AAET. Finally, section 6 summarises our main conclusions and outlines areas of future research.

[1] `www.pearsonpte.com/news/Pages/PTEAcademiclaunch.aspx`

[2] Williamson (2009) also states that e-Rater will also be used operationally from mid-2009 for assessing components of ETS's TOEFL exam, but in conjunction with human marking.

2 Technical Background

A full history of automated assessment is beyond the scope of this report. For recent reviews of work on automated essay or free-text assessment see Dikli (2006) and Williamson (2009). In this section, we focus on the ETS's e-Rater and PearsonKT's IEA systems as these are two of the three main systems which are operationally deployed. We do not consider IntelliMetric further as there is no precise and detailed technical description of this system in the public domain (Williamson, 2009). However, we do discuss a number of academic studies which assess and compare the performance of different techniques and as well as that of the public domain prototype system, BETSY (Rudner and Lang, 2002), which treats automated assessment as a Bayesian text classification problem, as this work sheds useful light on the potential of approaches other than those deployed by e-Rater and IEA.

2.1 e-Rater

e-Rater is extensively described in a number of publications and patents (e.g. Burstein, 2003; Attali and Burstein, 2006; Burstein *et al*, 2002, 2005). The most recently described version of e-Rater uses 10 broad feature types extracted from the text using NLP techniques, 8 represent writing quality and 2 content. These features correspond to high-level properties of a text, such as grammar, usage (errors), organisation or prompt/topic-specific content. Each of these high-level features is broken down into a set of ground features; for instance, grammar is subdivided into features which count the number of auxiliary verbs, complement clauses, and so forth, in a text. These features are extracted from the essay using NLP tools which automatically assign part-of-speech tags to words and phrases, search for specific lexical items, and so forth. Many of the feature extractors are manually written and based on essay marking rubrics used as guides for human marking of essays for specific examinations. The resulting counts for each feature are associated with cells of a vector which encodes all the grammar features of a text. Similar vectors are constructed for the other high-level features.

The feature extraction system outlined above, and described in more detail in the references provided, allows any text to be represented as a set of vectors each representing a set of features of a given high-level type. Each feature in each vector is weighted using a variety of techniques drawn from the fields of information retrieval (IR) and machine learning (ML). For instance, content-based analysis of an essay is based on vectors of individual word frequency counts drawn from text. Attali and Burstein (2006) transform frequency counts to weights by normalising the word counts to that of the most frequent word in a training set of manually-marked essays written in response to the same prompt, scored on a 6 point scale. Specifically, they remove stop words which are expected to occur with about equal frequency in all texts (such as *the*), then for each of the score points, the weight for word i at point p is:

$$W_{ip} = (\frac{F_{ip}}{MaxF_p}) * log(\frac{N}{N_i}) \quad (1)$$

where F_{ip} is the frequency of word i at score point p, $MaxF_p$ is the maximum frequency

of any word at scope point p, N is the total number of essays in the training set, and N_i is the total number of essays having word i in all score points in the training set [3]. For automated assessment of the content of an unmarked essay, this weighted vector is computed by dropping the conditioning on p and the result is compared to aggregated vectors for the marked training essays in each class using cosine similarity. The unmarked essay is assigned a content score corresponding to the most similar class. This approach transforms an unsupervised weighting technique, which only requires an unannotated collection of essays or documents, into a supervised one which requires a set of manually-marked prompt-specific essays.

Other vectors are weighted in different ways depending on the type of features extracted. Counts of grammatical, usage and style features are smoothed by adding 1 to all counts (avoiding zero counts for any feature), then divided by essay length word count to normalise for different essay lengths, then transformed to logs of counts to avoid skewing results on the basis of abnormally high counts for a given feature. Rhetorical organisation is computed by random indexing (Kanerva *et al*, 2000), a modification of latent semantic indexing (see section 2.2), which constructs word vectors based on cooccurrence in texts. Words can be weighted using a wide variety of weight functions (Gorman and Curran, 2006). Burstein *et al* (2005) describe an approach which calculates mean vectors for words from training essays which have been manually marked and segmented into passages performing different rhetorical functions. Mean vectors for each score point and passage type are normalised to unit length and transformed so they lie on the origin of a graph of the transformed geometric space. This controls for differing passage lengths and incorporates inverse document frequency into the word weights. The resulting passage vectors can now be used to compare the similarity of passages within and across essays, and, as above, to score essays for organisation via similarity to mean vectors for manually-marked training passages.

The set of high-level feature scores obtained for a given essay are combined to give an overall score. In earlier versions of e-Rater this was done by stepwise linear regression to assign optimal weights to the component scores so that the correlation with manually-assigned overall scores on the training set was maximised. However, Attali and Burstein (2006) advocate a simpler and more perspicuous approach using the weighted average of standardised feature scores, where the weights can be set by expert examiners based on marking rubrics. Williamson (2009) in his description of e-Rater implies a return to regression-based weighting.

2.2 Intelligent Essay Assessor (IEA)

The IEA like e-Rater assesses essays in terms of a small number of high-level features such as content, organisation, fluency, and grammar. The published papers and patent describing the techniques behind IEA (e.g. Landauer *et al*, 2000, 2003; Foltz *et al*, 2002)

[3] Burstein *et al* (2002) patent a different but related weighting of these counts using inverse document frequency (i.e. the well-known tf/idf weighting scheme introduced into IR by Sparck-Jones, 1972). Inverse document frequency is calculated from a set of prompt-specific essays which have been manually marked and assigned to classes. Presumably this was abandoned in favour of the current approach based on experimental comparison.

focus on the use of latent semantic analysis (LSA), a technique originally developed in IR to compute the similarity between documents or between documents and keyword queries by clustering words and documents so that the measurement of similarity does not require exact matches at the word level. For a recent tutorial introduction to LSA see Manning *et al* (2008:ch18). Landauer *et al* (2000) argue that LSA measures similarity of semantic content and that semantic content is dominant in the assessment of essays.

As for the content analysis component of e-Rater, LSA represents a document as a vector of words and requires a training set of prompt-specific manually-marked essays. However, instead of computing the cosine similarity directly between aggregated or mean vectors from the training set and the essay to be assessed, LSA deploys singular value decomposition (SVD) to reduce the dimensions of a matrix of words by essays to obtain a new matrix with reduced dimensions which effectively clusters words with similar contexts (i.e. their distribution across essays) and clusters essays with similar words. Words can be weighted to take account of their frequency in an essay and across a collection of essays before SVD is applied. LSA can be used to measure essay coherence as well by comparing passages within an essay and passages of similar rhetorical type from other essays. In this respect, there is little difference between e-Rater and IEA, because random indexing is simply a computationally efficient approximation of SVD which avoids construction of the full word-by-essay cooccurrence matrix.

Though it seems clear that IEA uses LSA to assess content and organisation (Foltz *et al*, 2002), it is unclear which other high-level features are computed this way. It is very unlikely that LSA is used to assess grammar or spelling, though there is no published description of how these features are assessed. On the other hand, it is likely that features like fluency are assessed via LSA, probably by using training annotated sets of text passages which illustrate this feature to different degrees so that a score can be assigned in the same manner as the content score. IEA, by default, combines the score obtained from each high-level feature into an overall score using multiple regression against human scores in a training set. However, this can be changed for specific examinations based, for example, on the marking rubric (Landauer *et al*, 2000).

2.3 Other Research on Automated Assessment

2.3.1 Text Classification

Both e-Rater and IEA implicitly treat automated assessment, at least partly, as a text classification problem. Whilst the roots of vector-based representations of text as a basis for measuring similarity between texts lie in IR (Salton, 1971), and in their original form can be deployed in an unsupervised fashion, their use in both systems is supervised in the sense that similarity is now measured relative to training sets of premarked essays (along several dimensions), and thus test essays can be classified on a grade point scale. Manning *et al* (2008:ch13) provides a tutorial introduction to text classification, an area of ongoing research which lies at the intersection of IR and ML and which has been given considerable impetus recently by new techniques emerging from ML.

Leakey (1998) explicitly modelled automated assessment as a text classification problem

comparing the performance of two standard classifiers, binomial Naive Bayes (NB) and kNN, over four different examination datasets. He found that binomial NB outperformed kNN and that the best document representation was a vector of lemmas or stemmed words rather than word forms. Rudner and Liang (2002) describe BETSY (the Bayesian Essay Test Scoring sYstem), which uses either a binomial or multinomial NB classifier and represents essays in terms of unigrams, bigrams and non-adjacent bigrams of word forms. A full tutorial on NB classifiers can be found in Manning *et al* (2008:ch13). Briefly, however, a multinomial model will estimate values for instances of the defined feature types from training data for each class type, which in the simplest case could be just 'pass' and 'fail', by smoothing and normalizing frequency counts for each feature, F, for example, for bigrams:

$$P(F_{bigram-i}) = \frac{Freq(W_j, W_k) + 1}{Freq(W_j) + N} \tag{2}$$

where N is the total bigram frequency count for this portion of the training data. To predict the most likely class for new unlabelled text, the log class-conditional probabilities of the features found in the new text are summed for each (C, e.g. pass/fail)

$$log(P(C)) + \sum_i log(P(F_i \mid C)) \tag{3}$$

and added to the prior probability of the class, typically estimated from the proportion of texts in each class in the training data. Taking the sum of the logs assumes ('naively') that each feature instance, whether unigram, bigram or whatever, is independent of the others. This is clearly incorrect, though suffices to construct an accurate and efficient classifier in many situations. In practice, within the NB framework, more sophisticated feature selection or weighting to handle the obvious dependencies between unigrams and bigrams would probably improve performance, as would adoption of a classification model which does not rely on such strong independence assumptions.

BETSY is freely available for research purposes. Coniam (2009) trained BETSY for AAET on a corpus of manually-marked year 11 Hong Kong ESOL examination scripts. He found that non-adjacent bigrams or word pairs provided the most useful feature types for accurate assessment. Both approaches use regression to optimise the fit between the output of the classifiers, which in the case of the Bayesian classifiers can be interpreted as the degree of statistical certainty or confidence in a given classification, to the grade point scales used in the different examinations.

Text classification is a useful model for automated assessment as it allows the problem to be framed in terms of supervised classification using machine learning techniques, and provides a framework to support systematic exploration of different classifiers with different representations of the text. From this perspective, the extant work has only scratched the surface of the space of possible systems. For instance, all the approaches discussed so far rely heavily on so-called 'bag-of-words' representations of the text in which positional and structural information is ignored, and all have utilised non-discriminative classifiers. However, there are strong reasons to think that, at least for AAET, grammatical competence and performance errors are central to assessment, but these are not captured well by a bag-of-words representation. In general discriminative classification techniques have

performed better on text classification problems than non-discriminative techniques, such as NB classifiers, using similar feature sets (e.g. Joachims, 1998), so it is surprising that discriminative models have not been applied to automated essay assessment.

2.3.2 Structural Information

Page (1994) was the first to describe a system which used partial parsing to extract syntactic features for automated assessment. e-Rater extended this work using hand coded extractors to look for specific syntactic constructions and specific types of grammatical error (see section 2.1 and references therein). Kanejiya *et al* (2003) describe an extension to LSA which constructs a matrix of words by essays in which words are paired with the part-of-speech tag of the previous word. This massively increases the size of the resultant matrix but does take account of limited structural information. However, their comparative experiments with pure LSA showed little improvement in assessment performance.

Lonsdale and Krause (2003) is the first application of a minimally-modified standard syntactic parser to the problem of automated assessment. They use the Link Parser (Sleator and Temperley, 1993) with some added vocabulary to analyse sentences in ESOL essays. The parser outputs the set of grammatical relations which hold between word pairs in the sentence, but also is able to skip words and output a cost vector (including the number of words skipped and the length of the sentence), when faced with ungrammatical input. The system scored essays by scoring each sentence on a five point scale, based on its cost vector, and then averaging these scores.

Rosé *et al* (2003) directly compare four different approaches to automated assessment on a corpus of physics essays. These are a) LSA over words, b) a NB text classifier over words, c) bilexical grammatical relations and syntactic features, such as passive voice, from sentence-by-sentence parses of the essays, and d) a model integrating b) and c). They found that the NB classifier outperformed LSA, whilst the model-based on parsing outperformed the NB classifier, and the model integrating parse information and the NB classifier performed best.

This body of work broadly supports the intuition that structural information is relevant to assessment but the only direct comparison of LSA, word-based classification and classification via structural information is on physics essays and may not, therefore, be comparable for ESOL. Lonsdale and Krause show reasonable correlation with human scoring using parse features alone on ESOL essays, but they conduct no comparative evaluation. The experimental design of Rosé *et al* is much better but a similar experiment needs to be conducted for ESOL essays.

2.3.3 Content Analysis

IEA exploits LSA for content analysis and e-Rater uses random indexing (RI). Both techniques are a form of word-by-essay clustering which allow the systems to generalise from specific to distributionally related words as measured by their occurrence in similar essays. However, there are many other published techniques for constructing distributional

semantic 'spaces' of this general type (see e.g. Turney and Pantel (2010) for a survey). Both probabilistic LSA (PLSA) and Latent Dirichlet Allocation (LDA) have been shown to work better than LSA for some IR applications. Kakkonen *et al* (2006) compare the performance of both to LSA on a corpus of graded Finnish essays on various topics. They found that LDA performed worse than LSA and that PLSA performed similarly.

There are many further possibilities in the area of content analysis that remain to be tried. Firstly, there are more recent approaches to constructing such distributional semantic spaces which have been shown to outperform RI and SVD-based techniques like LSA on the task of clustering words by semantic similarity, which is arguably central to the content analysis component of automated assessment. For example, Baroni *et al* (2007) show that Incremental Semantic Analysis (ISA) leads to better performance on semantic categorisation of nouns and verbs. ISA is an improvement of RI. Initially each word w is assigned a *signature*, a sparse vector, s_w, of fixed dimensionality d made up of a small number of randomly distributed +1 and -1 cells with all other cells assigned 0. d is typically much smaller than the dimensionality of the possible contexts (cooccurrences) of words given the text contexts used to define cooccurrence. At each occurrence of a target word t with a context word c, the history vector of t is updated as follows:

$$h_t += i(m_c h_c + (1 - m_c)s_c) \tag{4}$$

where i is a constant impact rate and m_c determines how much the history of one word influences the history of another word – the more frequent a context word the less it will influence the history of the target word. The m weight of c decreases as follows:

$$m_c = \frac{1}{exp(\frac{Freq(c)}{K_m})} \tag{5}$$

where K_m is a parameter determining rate of decay. ISA has the advantage that it is fully incremental, does not rely on weighting schemes that require global computations over contexts, and is therefore efficient to compute. It extends RI by updating the vector for t with the signature and history of c so that second order effects of the context word's distribution are factored into the representation of the target word.

As well as exploring improved clustering techniques over LSA or RI such as ISA, both the weighting functions used for modelling cooccurrence (e.g. Gorman and Curran, 2006), and the contexts used to assess cooccurrence (e.g. Baroni and Lenci, 2009), which has been exclusively based on an entire essay in automated assessment work, should be varied. For instance, the best models of semantic similarity often measure cooccurrence of words in local syntactic contexts, such as those provided by the grammatical relations output by a parser. Finally, though prompt-specific content analysis is clearly important for assessment of many essays types, it is not so clear that it is a central aspect of ESOL assessment, where demonstration of communicative competence and linguistic variety without excessive errors is arguably more important than the specific topic addressed.

2.4 Evaluation

The evaluation of automated assessment systems has largely been based on analyses of correlation with human markers. Typically, systems are trained on premarked essays for

a specific exam and prompt and their output scaled and fitted to a particular grade point scheme using regression or expert rubric-based weighting. Then the Pearson correlation coefficient is calculated for a set of test essays for which one or more human gradings are available. Using this measure, both e-Rater, IEA and other approaches discussed above have been shown to correlate well with human grades. Often they correlate as well as the grades assigned by two or more human markers on the same essays. Additionally, the rates of exact replication of human scores, of deviations by one point, and so forth can be calculated. These may be more informative about causes of larger divergences given specific phenomena in essays (e.g. Williamson, 2009; Coniam, 2009).

A weakness of the above approach is that it is clear that it is relatively easy to build a system that will correlate well with human markers under ideal conditions. Even the original PEG (Page, 1966) obtained high correlations using very superficial textual features such as essay, word and sentence length. However, such features are easily 'gamed' by students and by instructors 'teaching to the exam' (assessment regime) once it is public knowledge what features are extracted for automated assessment. As automated assessment is not based on a full understanding of an essay, the features extracted are to some extent proxies for such understanding. The degree to which such proxies can be manipulated independently of the features that they are intended to measure is clearly an important factor in the analysis of systems, especially if they are intended for use in high-stakes assessment. Powers *et al* (2002) conducted an experiment in which a variety of experts were invited to design and submit essays that they believed would either be under- or over-scored by e-Rater. The results showed that e-Rater was relatively robust to such 'gaming', though those with intimate knowledge of e-Rater were able to trick it into assigning scores deviating from human markers, even by 3 or more points on a 6-point scale.

A further weakness of comparison with human markers, and indeed with training such systems on raw human marks, is that human markers are relatively inconsistent and show comparatively poor correlation with each other. Alternatives, have been proposed such as training and/or testing on averaged or RASCH-corrected scores (e.g. Coniam, 2009), or evaluating by correlating system grades on one task, such essay writing, with human scores on an independent task, such as spoken comprehension (Attali and Burstein, 2006). Finally, many non-technical professionals involved in assessment object to automated assessment, arguing, for example, that a computer can never recognise creativity. In the end, this type of philosophical objection tends to dissipate as algorithms become more effective at any given task. For example, few argue that computers will never be able to play chess properly now that chess programs regularly defeat grand masters, though some will argue that prowess at chess is not in fact a sign of 'genuine intelligence'.

Nevertheless, it is clear that very thorough evaluation of assessment systems will be required before operational, especially high stakes, deployment and that this should include evaluation in adversarial scenarios and on unusual 'outlier' data, whether this be highly creative or deviant. From this perspective it is surprising that Powers *et al* (2002) is the sole study of this kind, though both e-Rater and IEA are claimed to incorporate mechanisms to flag such outliers for human marking.

3 AAET using Discriminative Preference Ranking

One of the key weaknesses of the text classification methods deployed so far for automated assessment is that they are based on non-discriminative machine learning models.

Non-discriminative models often embody incorrect assumptions about the underlying properties of the texts to be classified – for example, that the probability of each feature (e.g. word or ngram) in a text is independent of the others, in the case of the NB classifier (see section 2.3). Such models also weight features of the text in ways only loosely connected to the classification task – for example, possibly smoothed class conditional maximum likelihood estimates of features in the case of the NB classifier (see again section 2.3).

In this work, we apply discriminative machine learning methods, such as modern variants of the Large Margin Perceptron (Freund and Schapire, 1998) and the Support Vector Machine (SVM, Vapnik, 1995) to AAET. To our knowledge, this is the first such application to automated essay assessment. Discriminative classifiers make weaker assumptions concerning the properties of texts, directly optimize classification performance on training data, and yield optimal predictions if training and test material is drawn from the same distribution (see Collins (2002) for extended theoretical discussion and proofs).

In our description of the classifiers, we will use the following notation:

N	number of training samples
ν	avg. number of unique features / training sample
$\mathcal{X} \in \mathcal{P}R^D$	real D-dimensional sample space
$\mathcal{Y} = \{+1, -1\}$	binary target label space
$\mathbf{x}_i \in \mathcal{X}$	vector representing the ith training sample
$y_i \in \{+1, -1\}$	binary category indicator for ith training sample
$f : \mathcal{X} \rightarrow \mathcal{Y}$	classification function

3.1 Support Vector Machine

Linear SVMs (Vapnik, 1995) learn wide margin classifiers based on Structural Risk Minimization and continue to yield state-of-the-art results in text classification experiments (e.g. Lewis *et al*, 2004). In its dual form, linear SVM optimization equates to minimizing the following expression:

$$-\sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \tag{6}$$

subject to the constraint $\sum_i \alpha_i y_i = 0$ where the α's are the weight coefficients. The prediction is given by:

$$f(\mathbf{x}) = sign(\sum_i \alpha_i y_i \mathbf{x}_i \cdot \mathbf{x} + b) \tag{7}$$

where b is the bias and $sign(r) \in \{-1, +1\}$ depending on the sign of the input.

The practical use of the SVM model relies on efficient methods of finding approximate solutions to the quadratic programming (QP) problem posed by (6). A popular solution is

implemented in Joachims' SVM^{light} package (Joachims, 1999), in which the QP problem is decomposed into small constituent subproblems (the 'working set') and solved sequentially. This yields a training complexity at each iteration of $O(q^2 \cdot \nu)$ where q is the size of the working set. The efficiency of the procedure lies in the fact that $q \ll N$. The number of iterations is governed by the choice of q which makes it difficult to place a theoretical complexity bound on the overall optimization procedure, but experimental analysis by Yang *et al* (2003) suggests a super-linear bound of approximately $O(N^{1.5})$ with respect to the number of training samples, though in our experience this is quite heavily dependent on the separability of the data and the value of the regularization hyperparameter.

The per sample time complexity for prediction in the SVM model is $O(M \cdot \nu)$ where M is the number of categories, as a separate classifier must be trained for each category.

3.2 Timed Aggregate Perceptron

We now present a description of a novel variant of the batch perceptron algorithm, the Timed Aggregate Perceptron (TAP, Medlock, 2010). We will first introduce the ideas behind our model and then provide a formal description.

The online perceptron learning model has been a mainstay of artificial intelligence and machine learning research since its introduction by Rosenblatt (1958). The basic principle is to iteratively update a vector of weights in the sample space by adding some quantity in the direction of misclassified samples as they are identified. The *Perceptron with Margins* (PAM) was introduced by Krauth and Mezard (1987) and shown to yield better generalisation performance than the basic perceptron. More recent developments include the *Voted Perceptron* (Freund and Schapire, 1998) and the *Perceptron with Uneven Margins* (PAUM), applied with some success to text categorization and information extraction (Li *et al*, 2005).

The model we present is based on the *batch training* method (e.g. Bos and Opper, 1998) where the weight vector is updated in the direction of *all* misclassified instances simultaneously. In our model an *aggregate vector* is created at each iteration by summing all misclassified samples and normalising according to a *timing variable* which controls both the magnitude of the aggregate vector and the stopping point of the training process. The weight vector is then augmented in the direction of the aggregate vector and the procedure iterates. The timing variable is responsible for protection against overfitting; its value is initialised to 1, and gradually diminishes as training progresses until reaching zero, at which point the procedure terminates.

Given a set of N data samples paired with target labels $(\mathbf{x}_i, y_i)$ the TAP learning procedure returns an optimized weight vector $\hat{\mathbf{w}} \in \mathbf{R}^D$. The prediction for a new sample $\mathbf{x} \in \mathbf{R}^D$ is given by:

$$f(\mathbf{x}) = sign(\hat{\mathbf{w}} \cdot \mathbf{x}) \tag{8}$$

where the *sign* function converts an arbitrary real number to $+/-1$ based on its sign. The default decision boundary lies along the unbiased hyperplane $\hat{\mathbf{w}} \cdot \mathbf{x} = 0$, though a threshold can easily be introduced to adjust the bias.

At each iteration, an aggregate vector $\tilde{\mathbf{a}}_t$ is constructed by summing all misclassified

samples and normalising:

$$\tilde{\mathbf{a}}_t = norm(\sum_{\mathbf{x}_i \in Q_t} \mathbf{x}_i y_i \ , \tau) \tag{9}$$

$norm(\mathbf{a}, \tau)$ normalises $\mathbf{a}$ to magnitude τ and Q_t is the set of misclassified samples at iteration t, with the misclassification condition given by:

$$\mathbf{w}_t \cdot \mathbf{x}_i y_i < 1 \tag{10}$$

A margin of $+/-1$ perpendicular to the decision boundary is required for correct classification of training samples.

The timing variable τ is set to 1 at the start of the procedure and gradually diminishes, governed by:

$$\tau_t = \tau_{t-1} - \begin{cases} 0 & L_{t-1} > L_t \\ t(L_t - L_{t-1})\beta & \text{otherwise} \end{cases} \tag{11}$$

The class-normalised empirical loss, L_t, falls within the range $(0, 1)$ and is defined as:

$$L_t = \frac{1}{2}\left[\frac{|Q_t^+|}{N^+} + \frac{|Q_t^-|}{N^-}\right] \tag{12}$$

with $N^{+/-}$ denoting the number of class $+/-1$ training samples respectively. β is a measure of the balance of the training distribution sizes:

$$\beta = \frac{\min(N^+, N^-)}{N} \tag{13}$$

with an upper bound of 0.5 representing perfect balance. Termination occurs when either τ or the empirical loss reaches zero. How well the TAP solution fits the training data is governed by the rapidity of the timing schedule; earlier stopping leads to a more approximate fit.

In some cases, it may be beneficial to tune the rapidity of the timing schedule to achieve optimal performance on a specific problem, particularly when cross validation is feasible. In this instance we propose a modified version of expression (11) that includes a timing rapidity hyperparameter, r:

$$\tau_t = \tau_{t-1} - \begin{cases} r - 1 & L_{t-1} > L_t \\ rt(L_t - L_{t-1})\beta & \text{otherwise} \end{cases} \tag{14}$$

Note that this expression is equivalent to (11) in the case that $r = 1$.

An overview of the TAP learning procedure is given in Algorithm 1.

The timing mechanism used in our algorithm is motivated by the principle of *early stopping* in perceptron training (Bos and Opper, 1998), where the procedure is halted before reaching the point of minimum empirical loss. In our formulation, τ also governs the length of the aggregate vector, which is analogous to the learning rate in the standard perceptron. τ is decreased only when the class-normalised empirical loss increases. An increase in emprical loss is an indication either that the model is beginning to overfit or that the learning rate is too high, and a consequent decrease in τ works to counter both possibilities. The scale of the decrease is governed by three heuristic factors:

Algorithm 1 – TAP training procedure

Require: training data $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$
$\tau = 1$
for $t = 1, 2, 3 \ldots$ **do**
  if $\tau_t = 0 \vee L_t = 0$ **then**
    terminate and return $\mathbf{w}_t$
  else
    $\mathbf{w}_{t+1} = \mathbf{w}_t + \tilde{\mathbf{a}}_t$
  end if
  compute τ_{t+1}
end for

1. how far the algorithm has progressed (t)
2. the magnitude of the increase in empirical loss ($L_t - L_{t-1}$)
3. the balance of the training distributions (β)

The motivation behind the third heuristic is that in the early stages of the algorithm, unbalanced training distributions lead to aggregate vectors that are skewed toward the dominant class. If the procedure is stopped too early, the empirical loss will be disproportionately high for the subdominant class, leading to a skewed weight vector. The effect of β is to relax the timing schedule for imbalanced data which results in higher quality solutions.

The TAP optimisation procedure requires storage of the input vectors along with the feature weight and update vectors, yielding space complexity of $O(N)$ in the number of training samples. At each iteration, computation of the empirical loss and aggregate vector is $O(N \cdot \nu)$ (recall that ν is the average number of unique features per sample). Given the current and previous loss values, computing τ is $O(1)$ and thus each iteration scales with time complexity $O(N)$ in the number of training samples. The number of training iterations is governed by the rapidity of the timing schedule which has no direct dependence on the number of training samples, yielding an approximate overall complexity of $O(N)$ (linear) in the number of training samples.

3.3 Discriminative Preference Ranking

The TAP and SVM models described above perform binary discriminative classification, in which training exam scripts must be divided into 'pass' and 'fail' categories. The confidence margin generated by the classifier on a given test script can be interpreted as an estimate of the degree to which that script has passed or failed, e.g. a 'good' pass or a 'bad' fail. However, this gradation of script quality is not modelled explicitly by the classifier, rather it relies on emergent correlation of key features with script quality.

In this section, we introduce an alternative ML technique called preference ranking which is better suited to the AAET task. It explicitly models the relationships between scripts by learning an optimal ranking over a given sample domain, inferred through an optimisation

procecure that utilises a specified ordering on training samples. This allows us to model the fact that some scripts are 'better' than others, across an arbitrary grade range, without necessarily having to specify a numerical score for each, or introduce an arbitrary pass/fail boundary.

We now present a version of the TAP algorithm that efficiently learns preference ranking models. A derivation of similar equations for learning SVM-based models, and proof of their optimality is given by Joachims (2002).

The TAP preference ranking optimisation procedure requires a set of training samples, $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$, and a ranking $<_r$ such that the relation $\mathbf{x}_i <_r \mathbf{x}_j$ holds if and only if a sample $\mathbf{x}_j$ should be ranked higher than $\mathbf{x}_i$ for a finite, discrete partial or complete ranking or ordering, $1 \leq \ i, j \leq \ n, i \neq \ j$. Given some ranking $\mathbf{x}_i <_r \mathbf{x}_j$, the method only considers the difference between the feature vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ as evidence, known as *pairwise difference vectors*. The target of the optimisation procedure is to compute a weight vector $\hat{w}$ that minimises the number of margin-separated misranked pairs of training samples, as formalised by the following constraints on pairwise difference vectors:

$$\forall(\mathbf{x}_i <_r \mathbf{x}_j) : \hat{w} \cdot (\mathbf{x}_i - \mathbf{x}_j) \geq \mu. \tag{15}$$

where μ is the margin, given a specific value below.

The derived set of pairwise difference vectors grows quickly as a function of the number of training samples. An upper bound on the number of difference vectors for a set of training vectors is given by:

$$u = a^2 * r(r-1)/2 \tag{16}$$

where r is the number of ranks and a is the average rank frequency.

This yields intractable numbers of difference vectors for even modest numbers of training vectors, eg: $r = 4$, $a = 2000$ yields $24,000,000$ difference vectors.

To overcome this, the TAP optimisation procedure employs a sampling strategy to reduce the number of difference vectors to a manageable quantity. An upper bound is specified on the number of training vectors, and then the probability of sampling an arbitrary difference vector is given by u'/u where u' is the specified upper bound and u is given above.

The optimisation algorithm then proceeds as for the classification model (Algorithm 1), except we have a one-sided margin. The modified procedure is shown in Algorithm 2.

The misclassification condition is:

$$\mathbf{w}_t \cdot (\mathbf{x}_j - \mathbf{x}_i) > 2 \tag{17}$$

and the aggregate vector $\tilde{\mathbf{a}}_t$ is constructed by:

$$\tilde{\mathbf{a}}_t = norm(\sum_{\mathbf{x}_i <_r \mathbf{x}_j \in Q_t} \mathbf{x}_j - \mathbf{x}_i \ , \tau) \tag{18}$$

Algorithm 2 – TAP rank preference training procedure

Require: training data $\{(\mathbf{x}_1 <_r \mathbf{x}_2), \ldots, (\mathbf{x}_N <_r \mathbf{x}_{N+1})\}$
$\tau = 1$
for $t = 1, 2, 3 \ldots$ **do**
if $\tau_t = 0 \vee L_t = 0$ **then**
terminate and return $\mathbf{w}_t$
else
$\mathbf{w}_{t+1} = \mathbf{w}_t + \tilde{\mathbf{a}}_t$
end if
compute τ_{t+1}
end for

Note that the one-sided preference ranking margin takes the value 2, mirroring the two-sided unit-width margin in the classification model.

The termination of the optimisation procedure is governed by the timing rapidity hyper-parameter, as in the classification case, and training time is approximately linear in the number of pairwise difference vectors, upper bounded by u' (see above).

The output from the training procedure is an optimised weight vector $\mathbf{w}_t$ where t is the iteration at which the procedure terminated. Given a test sample, $\mathbf{x}$, Predictions are made, analogously to the classification model, by computing the dot-product $\mathbf{w}_t \cdot \mathbf{x}$. The resulting real scalar can then be mapped onto a grade/score range via simple linear regression (or some other procedure), or used in rank comparison with other test samples. Joachims (2002) describes an analogous procedure for the SVM model which we do not repeat here.

As stated earlier, in application to AAET, the principal advantage of this approach is that we explicitly model the grade relationships between scripts. Preference ranking allows us to model ordering in any way we choose; for instance we might only have access to pass/fail information, or a broad banding of grade levels, or we may have access to detailed scores. Preference ranking can account for each of these scenarios, whereas classification models only the first, and numerical regression only the last.

3.4 Feature Space

Intuitively AAET involves comparing and quantifying the linguistic variety and complexity, the degree of linguistic competence, displayed by a text against errors or infelicities in the performance of this competence. It is unlikely that this comparison can be captured optimally in terms of feature types like, for example, ngrams over word forms. Variety and complexity will not only be manifested lexically but also by the use of different types of grammatical construction, whilst grammatical errors of commission may involve non-local dependencies between words that are not captured by any given length of ngram. Nevertheless, the feature types used for AAET must be automatically extracted from text with good levels of reliability to be effectively exploitable.

We used the RASP system (Briscoe *et al* 2006; Briscoe, 2006) to automatically annotate

	Type	Example
	lexical terms	**and** / **mark**
	lexical bigrams	**dear_mary** / **of_the**
TFC:	part-of-speech tags	**NNL1** / **JJ**
	part-of-speech bigrams	**VBR_DA1** / **DB2_NN1**
	part-of-speech trigrams	**JJ_NNSB1_NP1** / **VV0_PPY_RG**
	parse rule names	**V1/modal_bse/+-** / **A1/a_inf**
TFS:	script length	*numerical*
	corpus-derived error rate	*numerical*

Table 1: Eight AAET Feature Types

both training and test data in order to provide a range of possible feature types and their instances so that we could explore their impact on the accuracy of the resulting AAET system. The RASP system is a pipeline of modules that perform sentence boundary detection, tokenisation, lemmatisation, part-of-speech (PoS) tagging, and syntactic analysis (parsing) of text. The PoS tagging and parsing modules are probabilistic and trained on native English text drawn from a variety of sources. For the AAET system and experiments described here we use RASP unmodified with default processing settings and select the most likely PoS sequence and syntactic analysis as the basis for feature extraction. The system makes availalble a wide variety of output representations of text (see Briscoe, 2006 for details). In developing the AAET system we experimented with most of them, but for the subset of experiments reported here we make use of the set of feature types given along with illustrative examples in Table 1.

Lower-cased but not lemmatised lexical terms (i.e. unigrams) are extracted along with their frequency counts, as in a standard 'bag-of-words' model. These are supplemented by bigrams of adjacent lexical terms. Unigrams, bigrams and trigrams of adjacent sequences of PoS tags drawn from the RASP tagset and most likely output sequence are extracted along with their frequency counts. All instances of these feature types are included with their counts in the vectors representing the training data and also in the vectors extracted for unlabelled test instances.

Lexical term and ngram features are weighted by frequency counts from the training data and then scaled using $tf \cdot idf$ weighting (Sparck-Jones, 1972) and normalised to unit length. Rule name counts, script length and error rate are linearly scaled so that their weights are of the same order of magnitude as the scaled term/ngram counts.

Parse rule names are extracted from the phrase structure tree for the most likely analysis found by the RASP parser. For example, the following sentence from the training data, *Then some though occured to me.*, receives the analysis given in Figure 1, whilst the corrected version, *Then a thought occurred to me.* receives the analysis given in Figure 2. In this representation, the nodes of the parse trees are decorated with one of about 1000 rule names, which are semi-automatically generated by the parser and which encode quite detailed information about the grammatical constructions found. However, in common with ngram features, these rule names are extracted as an unordered list from the analyses for all sentences in a given script along with their frequency counts. Each rule name

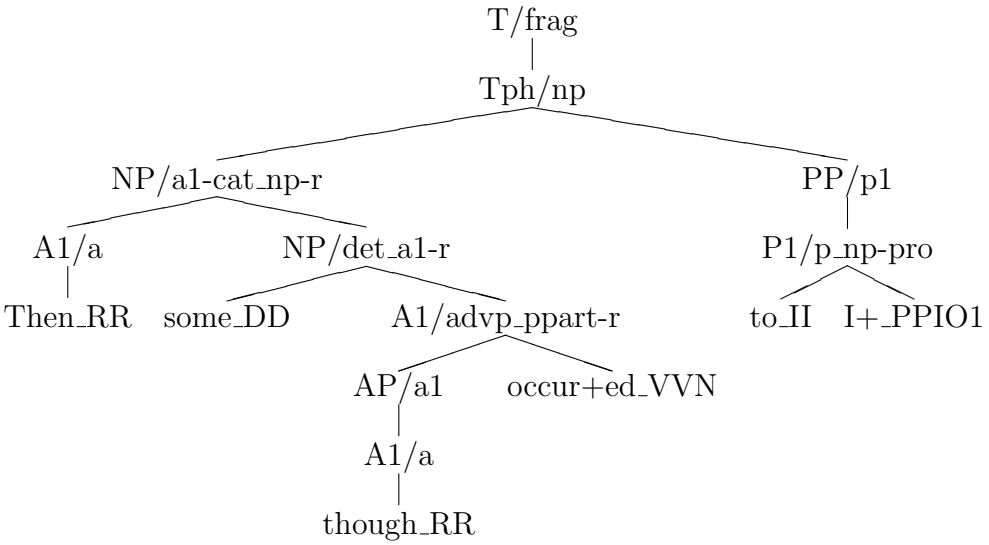

Figure 1: *Then some though occured to me*

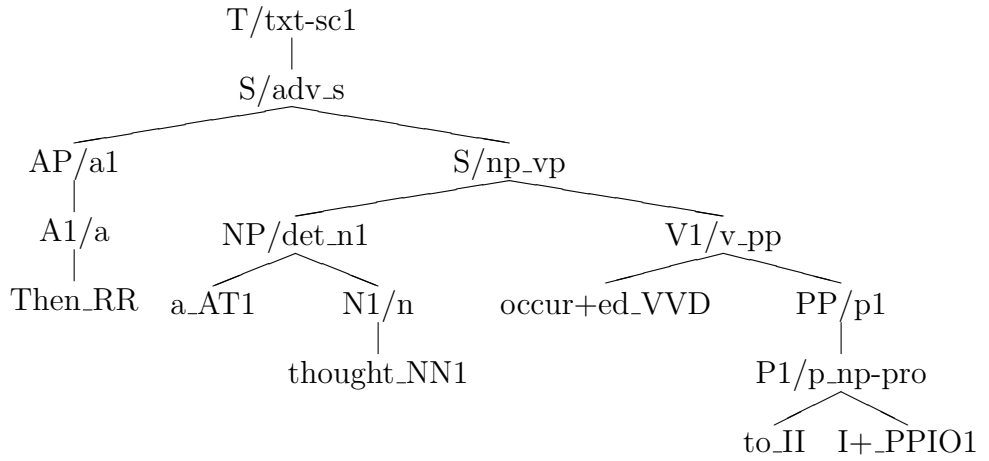

Figure 2: *Then a thought occurred to me*

together with its frequency count is represented as a cell in the vector derived from a script. The script length in words is used as a feature less for its intrinsic informativeness than for the need to balance the effect of script length on other features. For example, error rates, ngram frequencies, etc will tend to rise with the amount of text, but the overall quality of a script must be assessed as a ratio of the opportunities afforded for the occurrence of some feature to its actual occurrence.

The automatic identification of grammatical and lexical errors in text is far from trivial (Andersen, 2010). In the existing systems reviewed in section 2, a few specific types of well-known and relatively frequent errors, such as subject-verb agreement, are captured explicitly via manually-constructed error-specific feature extractors. Otherwise, errors are captured implicitly and indirectly, if at all, via unigram or other feature types. Our AAET system already improves on this approach because the RASP parser rule names explicitly represent marked, peripheral or rare constructions using the '-r' suffix, as well

as combinations of extragrammatical subsequences suffixed 'frag', as can be seen by comparing Figure 2 and Figure 1. These cues are automatically extracted without any need for error-specific rules or extractors and can capture many types of long-distance grammatical error. However, we also include a single numerical feature representing the overall error rate of the script. This is estimated by counting the number of unigrams, bigrams and trigrams of lexical terms in a script that do not occur in a very large 'background' ngram model for English which we have constructed from approximately 500 billion words of English sampled from the world wide web. We do this efficiently using a Bloom Filter (Bloom, 1970). We have also experimenented with using frequency counts for smaller models and measures such as mutual information (e.g. Turney and Pantel, 2010). However, the most effective method we have found is to use simple presence/absence over a very large dataset of ngrams which unlike, say, the Google ngram corpus (Franz and Brants, 2006) retains low frequency ngrams.

Although we have only described the feature types that we used in the experiments reported below, because they proved useful with respect to the competence level and text types investigated, it is likely that others made available by the RASP system, such as the connected, directed graph of grammatical relations over sentences, the degree of ambiguity within a sentence, the lemmas and/or morphological complexity of words, and so forth (see Briscoe 2006 for a fuller description of the range of feature types, in principle, made available by RASP), will be discriminative in other AAET scenarios. The system we have developed includes automated feature extractors for most types of feature made available through the various representations provided by RASP. This allows the rapid and largely automated discovery of an appropriate feature set for any given assessment task, using the experimental methodology exemplified in the next section.

4 The FCE Experiments

4.1 Data

For our experiments we made use of a set of transcribed handwritten scripts produced by candidates taking the First Certificate in English (FCE) examination written component. These were extracted from the Cambridge Learner Corpus (CLC) developed by Cambridge University Press. These scripts are linked to metadata giving details of the candidate, date of the exam, and so forth, as well as the final scores given for the two written questions attempted by candidates (see Hawkey, 2009 for details of the FCE). The marks assigned by the examiners are postprocessed to identify outliers, sometimes second marked, and the final scores are adjusted using RASCH analysis to improve consistency. In addition, the scripts in the CLC have been manually error-coded using a taxonomy of around 80 error types providing corrections for each error. The errors in the example from the previous section are coded in the following way:

```
<RD>some|a</RD> <SX>though|thought</SX> <IV>occured|occurred</IV>
```

where RD denotes a determiner replacement error, SX a spelling error, and IV a verb inflection error (see Nicholls 2003 for full details of the scheme). In our experiments, we

used around three thousand scripts from examinations set between 1997 and 2004, each about 500 words in length. A sample script is provided in the appendix.

In order to obtain an upper bound on examiner agreement and also to provide a better benchmark to assess the performance of our AAET system compared to that of human examiners (as recommended by, for example, Attali and Bernstein, 2006), Cambridge ESOL arranged for four senior examiners to remark 100 FCE scripts drawn from the 2001 examinations in the CLC using the marking rubric from that year. We know, for example, from analysis of these marks and comparison to those in the CLC that the correlation between the human markers and the CLC scores is about .8 (Pearson) or .78 (Spearman's Rank), thus establishing an upper bound for performance of any classifier trained on this data (see section 4.3 below).

4.2 Binary Classification

In our first experiment we trained five classifier models on 2973 FCE scripts drawn from the years 1999–2003. The aim was to apply well- known classification and evaluation techniques to explore the AAET task from a discriminative machine learning perspective and also to investigate the efficacy of individual feature types. We used the feature types described in section 3.4 with all the models and divided the training data into pass (mark above 23) and fail classes. Because there was a large skew in the training classes, with about 80% of the scripts falling into the pass class, we used the Break Even Precision (BEP) measure, defined as the point at which average precision=recall, (e.g. Manning *et al*, 2008) to evaluate the performance of the models on this binary classification task. This measure favours a classifer which locates the decision boundary between the two classes in such a way that false positives / negatives are evenly distributed between the two classes.

The models trained were naive Bayes, Baysian logistic regression, maximum entropy, SVM, and TAP. Consistent with much previous work on text classification tasks, we found that the TAP and SVM models performed best and did not yield significantly different results. For brevity, and because TAP is faster to train, we report results only for this model in what follows.

Figure 3 shows the contribution of feature types to the overall accuracy of the classifier. With unigram terms alone it is possible to achieve a BEP of 66.4%. The addition of bigrams of terms improves performance by 2.6% (representing about 19% relative error reduction (RER) on the upper bound of 80%). The addition of an error estimate feature based on the Google ngram corpus further improves performance by 2.9% (further RER about 21%). Addition of parse rule name features further improves performance by 1.5% (further RER about 11%). The remaining feature types in Table 1 contribute another 0.4% improvement (further RER about 3%).

These results provide some support for the choice of feature types described in section 3.4. However, the final datapoint in the graph in Figure 3 shows that if we substitute the error rate predicted from the CLC manual error coding for our corpus derived estimate, then performance improves a further 2.9%, only 3.3.% below the upper bound defined by the degree of agreement between human markers. This strongly suggests that the error

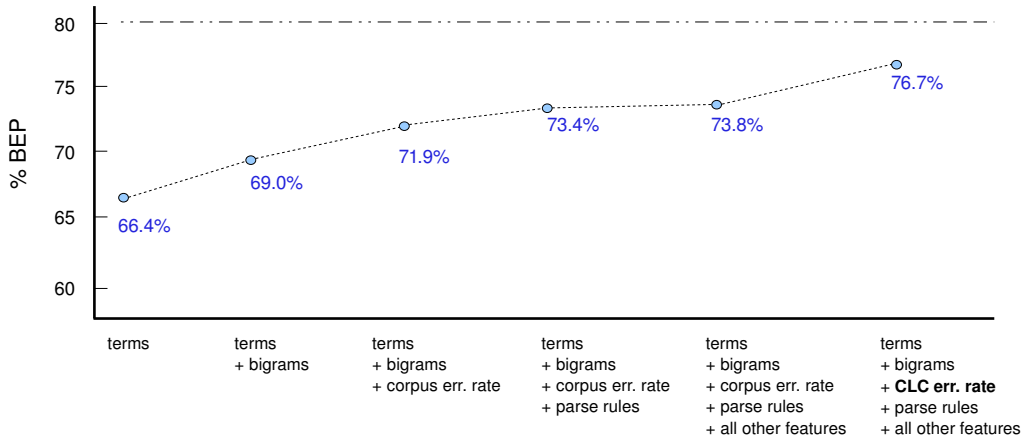

Figure 3: Contribution of Feature Types

estimate is a critical feature and that there is room for significant improvement in its method of estimation (see section 4.5 below).

4.3 Preference Ranking

As discussed earlier, a better way of modelling the AAET task is to use the preference ranking paradigm. Here we describe experiments carried out using the TAP implementation described in section 3.3.

We trained the TAP preference ranking model with the feature types described in section 3.4 as described in section 3.3, and compared the correlation of the predicted values with that of the four senior examiners and with the CLC scores for the test data. The results are given using Pearson's correlation coefficient in Table 2 and using Spearman's Rank correlation in Table 3.

	CLC	**Rater 1**	**Rater 2**	**Rater 3**	**Rater 4**	**Auto-mark**
CLC		0.82	0.77	0.73	0.78	0.78
Rater 1	0.82		0.85	0.85	0.88	0.79
Rater 2	0.77	0.85		0.78	0.79	0.77
Rater 3	0.73	0.85	0.78		0.78	0.76
Rater 4	0.78	0.88	0.79	0.78		0.71
Auto-mark	0.78	0.79	0.77	0.76	0.71	
Average:	0.77	0.84	0.79	0.78	0.79	0.76

Table 2: Correlation (Pearson's CC)

We also compared the performance of the preference ranking TAP model to a binary TAP classifier trained using the same feature types on the same data divided into pass/fail scripts. The correlation with the CLC scores on this test data was worse by 0.05 (Pearson) and 0.07 (Spearman) using classification as compared to preference ranking with the same underlying TAP model.

	CLC	Rater 1	Rater 2	Rater 3	Rater 4	Auto-mark
CLC		0.80	0.79	0.75	0.76	0.80
Rater 1	0.80		0.81	0.81	0.85	0.74
Rater 2	0.79	0.81		0.75	0.79	0.75
Rater 3	0.75	0.81	0.75		0.79	0.75
Rater 4	0.76	0.85	0.79	0.79		0.73
Auto-mark	0.80	0.74	0.75	0.75	0.73	
Average:	0.78	0.80	0.78	0.77	0.78	0.75

Table 3: Correlation (Spearman's Rank)

These results suggest that the AAET system we have developed is able to achieve levels of correlation similar to those achieved by the human markers both with each other and with the RASCH-adjusted marks in the CLC. To give a more concrete idea of the actual marks assigned and their variation, we give marks assigned to a random sample of 10 scripts from the test data in Table 4 (fitted to the appropriate score range by simple linear regression).

Auto-mark	Rater 1	Rater 2	Rater 3	Rater 4
26	26	23	25	23
33	36	31	38	36
29	25	22	25	27
24	23	20	24	24
25	25	22	24	22
27	26	23	30	24
5	12	5	12	17
29	30	25	27	27
21	24	21	25	19
23	25	22	25	25

Table 4: Sample predictions (random ten)

4.4 Temporal Sensitivity

The training data we have used so far in our experiments is drawn from examinations both before and after the test data. In order to investigate both the effect of different amounts of training data and also the effect of training on scripts drawn from examinations at increasing temporal distance from the test data, we divided the data by year and trained and tested the correlation (Pearson) with the CLC marks. Figure 4 shows the results – clearly there is an effect of training data size, as no result is as good as those reported using the full dataset for training. However, there is also a strong effect for temporal distance between training and test data, reflecting the fact that both the type of prompts used to elicit text and the marking rubrics evolve over time (e.g. Hawkey, 2009; Cecil and Weir, 2007).

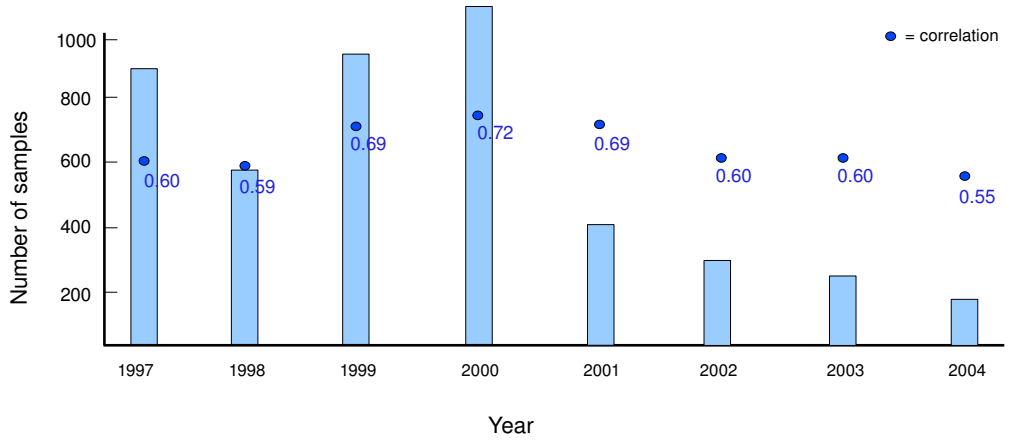

Figure 4: Training Data Effects

4.5 Error Estimation

In order to explore the effect of different datasets on the error prediction estimate, we have gathered a large corpus of English text from the web. Estimating error rate using a 2 billion word sample of text sampled from the UK domain retaining low frequency unigrams, bigrams, and trigrams we were able to improve performance over estimation using the Google ngram corpus by 0.09% (Pearson) in experiments which were otherwise identical to those reported in section 4.3

To date we have gathered about a trillion words of sequenced text from the web. We expect future experiments with error estimates based on larger samples of this corpus to improve on these results further. However the results reported here demonstrate the viability of this approach, in combination with parser-based features which implicitly capture many types of longer distance gramatical error, compared to the more labour intensive one of manually coding feature extractors for known types of stereotypical learner error.

4.6 Incremental Semantic Analysis

Although, the focus of our experiments has not been on content analysis (see section 2.3.3), we have undertaken some limited experiments to compare the performance of an AAET system based primarily on such techniques (such as PearsonKT's, IEA, see section 2) to that of the system presented here.

We used ISA (see section 2.3.3) to construct a system which, like IEA, uses similarity to an average vector constructed using ISA from high scoring FCE training scripts as the basis for assigning a mark. The cosine similarity scores were then fitted to the FCE scoring scheme. We trained on about a thousand scripts drawn from 1999 to 2004 and tested on the standard test set from 2001. Using this approach we were only able to obtain a correlation of 0.45 (Pearson) with the CLC scores and and average of 0.43 (Pearson) with the human examiners. This contrasts with scores of 0.47 (Pearson) and 0.45 (Pearson)

training the TAP ranked preference classifier on a similar number of scripts and using only unigram term features.

These results, taken with those reported above, suggest that there isn't a clear advantage to using techniques that cluster terms according to their context of occurrence, and compute text similarity on the basis of these clusters, over the text classification approach deployed here. Of course, this experiment does not demonstrate that clustering techniques cannot play a useful role in AAET, however, it does suggest that a straightforward application of latent or distributional semantic methods to AAET is not guaranteed to yield optimal results.

4.7 Off-Prompt Essay Detection

As discussed in section 2.4, one issue with with the deployment of AAET for high stakes examinations or other 'adversarial' contexts is that a non-prompt specific approach to AAET is vulnerable to 'gaming' via submission of linguistically excellent rote-learned text regardless of the prompt. To detect such off-prompt text automatically does require content analysis of the type discussed in section 2.3.3 and explored in the previous section as an approach to grading.

Given that our approach to AAET is not prompt-specific in terms of training data, ideally we would like to be able to detect off-prompt scripts with a system that doesn't require retraining for different prompts. We would like to train a system which is able to compare the question and answer script within a generic distributional semantic space. Because the prompts are typically quite short we cannot expect that in general there will be much direct overlap between contentful terms or lemmas in the prompt and those in the answer text.

We trained an ISA model using 10M words of diverse English text using a 250-word stop list and ISA parameters of 2000 dimensions, impact factor 0.0003, and decay constant 50 with a context window of 3 words. Each question and answer is represented by the sum of the history vectors corresponding to the terms they contain. We also included additional dimensions representing actual terms in the overall model of distributional semantic space to capture cases of literal overlap between terms in questions and in answers. The resulting vectors are then compared by calculating their cosine similarity. For comparison, we built a standard vector space model that measures semantic similarity using cosine distance between vectors of terms for question and answer via literal term overlap.

To test the performance of these two approaches to off-prompt essay detection, we extracted 109 passing FCE scripts from the CLC answering four different prompts:

1. During your holiday you made some new friends. Write a letter to them saying how you enjoyed the time spent with them and inviting them to visit you.

2. You have been asked to make a speech welcoming a well-known writer who has come to talk to your class about his/her work. Write what you say.

3. "Put that light out!" I shouted. Write a story which begins or ends with these words.

4. Many people think that the car is the greatest danger to human life today. What do you think?

Each system was used to assign each answer text to the most similar prompt. The accuracy (ratio of correct to all assignments) of of the standard vector space model was 85%, whilst the augmented ISA model achieved 93%. This preliminary experiment suggests that a generic model for flagging putative off-prompt essays for manual checking could be constructed by manual selection of a set of prompts from past papers and the current paper and then flagging any answers that matched a past prompt better than the current prompt. There will be some false positives, but these initial results suggest that an augmented ISA model could perform well enough to be useful. Further experimentation on larger sets of generic training text and on optimal tuning of ISA parameters may also improve accuracy.

5 Conclusions

In this report, we have introduced the discriminative TAP preference ranking model for AAET. We have demonstrated that this model can be coupled with the RASP text processing toolkit allowing fully automated extraction of a wide range of feature types many of which we have shown experimentally are discriminative for AAET. We have also introduced a generic and fully automated approach to error estimation based on efficient matching of text sequences with a very large background ngram corpus derived from the web using a Bloom filter, and have shown experimentally that this is the single most discriminative feature in our AAET model. We have also shown experimentally that this model performs significantly better than an otherwise equivalent one based on classification as opposed to preference ranking. We have also shown experimentally that text classification is at least as effective for AAET as a model based on ISA, a recent and improved latent or distributional semantic content-based text similarity method akin to that used in IEA. However, ISA is useful for detecting off-prompt essays using a generic model of distributional semantic space that does not require retraining for new prompts.

Much further work remains to be done. We believe that the features assessed by our AAET model make subversion by students difficult as they more directly assess linguistic competence than previous approaches. However, it remains to test this experimentally. We have shown that error estimation against a background ngram corpus is highly informative, but our fully automated technique still lags error estimates based on the manual error coding of the CLC. Further experimentation with larger background corpora and weighting of ngrams on the basis of their frequency, pointwise mutual information, or similar measures may help close this gap. Our AAET model is not trained on prompt-specific data, which is operationally advantageous, but it does not include any mechanism for detecting text lacking overall inter-sentential coherence. We believe that ISA or other recent distributional semantic techniques provide a good basis for adding such features to the model and plan to test this experimentally. Finally our current AAET system simply returns a score, though implicit in its computation is the identification of both negative and positive features that contribute to its calculation. We plan to explore methods for automatically providing feedback to students based on these features in order to facilitate

deployment of the system for self-assessment and self-tutoring.

In the near future, we intend to release a public-domain training set of anonymised FCE scripts from the CLC together with an anonymised version of the test data described in section 4. We also intend to report the performance of preference ranking with the SVM^{light} package (Joachims, 1999) based on RASP-derived features, and error estimation using a public domain corpus trained and tested on this data and compared to the performance of our best TAP-based model. This will allow better replication of our results and facilitate further work on AAET.

Acknowledgements

The research and experiments reported here were partly funded through a contract to iLexIR Ltd from Cambridge ESOL, a division of Cambridge Assessment, which in turn is a subsidiary of the University of Cambridge. We are grateful to Cambridge University Press for permission to use the subset of the Cambridge Learners' Corpus for these experiments. We are also grateful to Cambridge Assessment for arranging for the test scripts to be remarked by four of their senior examiners to facilitate their evaluation.

References

Andersen, O. (2010) *Grammatical error prediction,* Cambridge University, Computer Laboratory, PhD Dissertation.

Attali, Y. and Burstein, J. (2006) 'Automated essay scoring with e-rater v2', *Journal of Technology, Learning and Assessment, vol.4(3),*

Baroni, M., and Lenci, I. (2009) 'One distributional memory, many semantic spaces', *Proceedings of the Wkshp on Geometrical Models of Natural Language Semantics,* Eur. Assoc. for Comp. Linguistics, pp. 1–8.

Bos, S. and Opper, M. (1998) 'Dynamics of batch training in a perceptron', *J. Physics A: Math. & Gen., vol.31(21),* 4835–4850.

Burstein, J. (2003) 'The e-rater scoring engine: automated essay scoring with natural language processing' in (eds) Shermis, M.D. and J. Burstein (eds.), *Automated Essay Scoring: A cross-Disciplinary Perspective,* Lawrence Erlbaum Associates Inc., pp. 113–122.

Burstein, J., Braden-Harder, L., Chodorow, M.S., Kaplan, B.A., Kukich, K., Lu, C., Rock, D.A., and Wolff, S. (2002) *System and method for computer-based automatic essay scoring,* US Patent 6,366,759, April 2.

Burstein, J., Higgins, D., Gentile, C., and Marcu, D. (2005) *Method and system for determining text coherence,* US Patent 2005/0143971 A1, June 30.

Collins, M. (2002) 'Discriminative training methods for hidden Markov models: theory and experiments with Perceptron algorithms', *Proceedings of the Empirical Methods in Nat. Lg. Processing (EMNLP),* Assoc. for Comp. Linguistics, pp. 1–8.

Coniam, D. (2009) 'Experimenting with a computer essay-scoring program based on ESL student writing scripts', *ReCALL, vol.21(2),* 259–279.

Dikli, S. (2006) 'An overview of automated scoring of essays', *Journal of Technology, Learning and Assessment, vol.5(1),*

Elliot, S. (2003) 'Intellimetric^{TM}: From Here to Validity' in (eds) Shermis, M.D. and J. Burstein (eds.), *Automated Essay Scoring: A cross-Disciplinary Perspective,* Lawrence Erlbaum Associates Inc., pp. 71–86.

Foltz, P.W., Landauer, T.K., Laham, R.D., Kintsch, W., and Rehder, R.E. (2002) *Methods for analysis and evaluation of the semantic content of a writing based on vector length,* US Patent 6,356,864 B1, March 12.

Franz, A. and Brants, T. (2006) *All our N-gram are Belong to You,* http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html.

Freund, Y. and Schapire, R. (1998) 'Large margin classification using the perceptron algorithm', *Computational Learning Theory, vol.209–217,*

Gorman, J. and Curran, J.R. (2006) 'Random indexing using statistical weight functions', *Proceedings of the Conf. on Empirical Methods in Nat. Lg. Proc.,* Assoc. for Comp. Linguistics, pp. 457–464.

Hawkey, R. (2009) *Examining FCE and CAE: Studies in Language Testing, 28,* Cambridge University Press.

Joachims, T. (1998) 'Text categorization with support vector machines: learning with many relevant features', *Proceedings of the Proc. of Eur. Conf. on Mach. Learning,* Springer-Verlag, pp. 137–142.

Joachims, T. (1999) 'Making large-scale support vector machine learning practical' in (eds) Scholkopf, S.B. and C. Burges (eds.), *Advances in kernel methods,* MIT Press.

Joachims, T. (2002) 'Optimizing search engines using clickthrough data', *Proceedings of the SIGKDD,* Assoc. Computing Machinery.

Kakkonen, T., Myller, N., Sutinen, E. (2006) 'Applying Latent Dirichlet Allocation to automatic essay grading', *Proceedings of the FinTAL,* Springer-Verlag, pp. 110–120.

Kanejiya, D., Kamar, A. and Prasad, S. (2003) 'Automatic Evaluation of Students' Answers using Syntactically Enhanced LSA', *Proceedings of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing,* Assoc. for Comp. Linguistics.

Kanerva, P., Kristofersson, J., and Holst, A. (2000) 'Random indexing of text samples for latent semantic analysis', *Proceedings of the 22nd Annual Conf. of the Cognitive Science Society,* Cognitive Science Soc..

Krauth, W. and Mezard, M. (1987) 'Learning algorithms with optimal stability in neural networks', *J. of Physics A; Math. Gen., vol.20,*

Kukich, K. (2000) 'Beyond automated essay scoring' in (ed.) Hearst, M. (eds.), *The debate on automated essay grading,* IEEE Intelligent Systems, pp. 27–31.

Landauer, T.K., Laham, D., and Foltz, P.W. (2000) 'The Intelligent Essay Assessor', *IEEE Intelligent Systems, vol.15(5),*

Landauer, T.K., Laham, D. and Foltz, P.W. (2003) 'Automated scoring and annotation of essays with the Intelligent Essay Assessor' in (eds) Shermis, M.D. and J. Burstein (eds.), *Automated Essay Scoring: A cross-Disciplinary Perspective,* Lawrence Erlbaum Associates Inc., pp. 87–112.

Leakey, L.S. (1998) 'Automatic essay grading using text categorization techniques', *Proceedings of the 21st ACM-SIGIR,* Assoc. for Computing Machinery.

Lewis, D.D., Yang, Y., Rose, T. and Li, T. (2004) 'RCv1: A new benchmark collection for text categorization research', *J. Mach. Learning res., vol.5,* 361–397.

Li, Y., Bontcheva, K. and Cunningham, H. (2005) 'Using uneven margins svm and perceptron for information extraction', *Proceedings of the 9th Conf. on Nat. Lg. Learning,* Assoc. for Comp. Ling..

Lonsdale, D. and Strong-Krause, D. (2003) 'Automated Rating of ESL Essays', *Proceedings of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing,* Assoc. for Comp. Linguistics.

Manning, C., Raghavan, P.,and Schutze, H. (2008) *Introduction to Information Retrieval,* Cambridge University Press.

Nicholls, D. (2003) 'The Cambridge Learner Corpus: Error coding and analysis for lexicography and ELT' in Corpus Linguistics II (eds.), *Archer, D, Rayson, P., Wilson, A. and McCenery T. (eds.),* UCREL Technical Report 16, Lancaster University.

Page, E.B. (1966) 'The imminence of grading essays by computer', *Phi Delta Kappan, vol.48,* 238–243.

Page, E.B. (1994) 'Computer grading of student prose, using modern concepts and software', *Journal of Experimental Education, vol.62(2),* 127–142.

Powers, D.E., Burstein, J., Chodorow, M., Fowles, M.E., Kukich, K. (2002) 'Stumping e-rater: challenging the valdity of automated essay scoring', *Computers in Human Behavior, vol.18,* 103–134.

Rosenblatt, F. (1958) 'The perceptron: A probabilistic model for information storage and organization in the brain', *Psychological Review, vol.65,*

Rosé, C.P., Roque, A., Bhembe, D. and VanLehn, K. (2003) 'A Hybrid Text Classification Approach for Analysis of Student Essays', *Proceedings of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing,* Assoc. for Comp. Linguistics.

Rudner, L.M. and Lang, T. (2002) 'Automated essay scoring using Bayes' theorem', *Journal of Technology, Learning and Assessment, vol.1(2),*

Shaw, S and Weir, C. (2007) *Examining Writing in a Second Language, Studies in Language Testing 26,* Cambridge University Press.

Sparck Jones, K. (1972) 'A statistical interpretation of term specificity and its application in retrieval', *Journal of Documentation, vol.28(1),* 11–21.

Sleator, D. and Temperley, D. (1993) 'Parsing English with a Link Grammar', *Proceedings of the 3rd Int. Wkshp on Parsing Technologies,* Assoc. for Comp. Ling..

Turney, P. and Pantel, P. (2010) 'From frequency to meaning', *Jnl. of Artificial Intelligence Research, vol.37,* 141–188.

Vapnik, V.N. (1995) *The nature of statistical learning theory,* Springer-Verlag.

Williamson, D.M. (2009) *A framework for implementing automated scoring,* Educational Testing Service, Technical Report.

Yang, Y., Zhang, J. and Kisiel, B. (2003) 'A scalability analysis of classifiers in text categorization', *Proceedings of the 26th ACM-SIGIR,* Assoc. for Computing Machinery, pp. 96–103.

Appendix: Sample Script

The following is a sample of a FCE script with error annotation drawn from the CLC and converted to XML. The full error annotation scheme is described in Nicholls (2003).

<head title="lnr:1.01" entry="0" status="Active" url="571574"
sortkey="AT*040*0157*0100*2000*01">
<candidate>
<exam>
<exam_code>0100</exam_code>
<exam_desc>First Certificate in English</exam_desc>
<exam_level>FCE</exam_level></exam>
<personnel>
<ncode>011</ncode>
<language>German</language>
<age>18</age>
<sex>M</sex></personnel>
<text>
<answer1>
<question_number>1</question_number>
<exam_score>34.2</exam_score>
<coded_answer>
Dear Mrs Ryan<NS type="MP">|,</NS> Many
thanks for your letter. I would like to travel in July
because I have got <NS type="MD">|my</NS> summer holidays from July to
August and I work as a bank clerk in August. I think a tent would suit
my personal <NS type="RP">life-style|lifestyle</NS> better than a log
cabin because I love <NS type="UD">the</NS> nature. I
would like to play basketball during my holidays at Camp California
because I love this game. I have been playing basketball for 8 years and
today I am a member of an Austrian <NS type="RP">basketball-team|
basketball team</NS>. But I have never played golf in my life <NS
type="RC">but|though</NS> with your help I would be able to learn how to
play golf and I think this could be very interesting. <p
idx="15580">I <NS type="W">also would|would also</NS> like to know how
much money I will get from you for <NS type="RA">those|these</NS> two
weeks because I would like to spend some money <NS type="RT">for|on</NS>
clothes. I am looking forward to hearing from you
soon. Yours sincerely
</coded_answer></answer1>
<answer2>
<question_number>4</question_number>
<exam_score>30.0</exam_score>
<coded_answer>
Dear Kim Last month I enjoyed helping
at a pop concert and I think you want to hear some funny stories about
the <NS type="FN">experience|experiences</NS> I <NS type="RV">made|
had</NS>. At first I had to clean the three private
rooms of the stars. This was very boring but after I left the third room
I met Brunner and Brunner. These two people are stars in our country...
O.K. I am just <NS type="IV">kiding|kidding</NS>. I don't like <NS
type="W">the songs of Brunner and Brunner|Brunner and Brunner's
songs</NS> because this kind of music is very boring. <p
idx="15586">I also had to clean the <NS type="RN">washing rooms|

washrooms</NS>. I will never ever help anybody to <NS type="S">organice|
organise</NS> a pop concert <NS type="MY">|again</NS>. <p
idx="15587">But after this <NS type="S">serville|servile</NS> work I met
Eminem. I think you know his popular songs like "My Name Is". It was one
of the greatest moments in my life. I had to <NS type="RV">bring|
take</NS> him something to eat. It was <NS
type="UD">a</NS> hard but also <NS type="UD">a</NS> <NS type="RJ">funny|
fun</NS> work. You should try to <NS type="RV"><NS type="FV">called|
call</NS>|get</NS> some experience <NS type="RT">during|at</NS> such a
concert<NS type="RP"> you|. You</NS> would not regret it. <p
idx="15589">I am looking forward to hearing from you soon.
</coded_answer></answer2></text></head>