Grammatical Error Detection for Corrective Feedback Provision in Oral Conversations

Sungjin Lee, Hyungjong Noh, Kyusong Lee, Gary Geunbae Lee

Department of Computer Science and Engineering,
Pohang University of Science and Technology (POSTECH), South Korea
{junion, nohhj, kyusonglee, gblee}@postech.ac.kr

Abstract

The demand for computer-assisted language learning systems that can provide corrective feedback on language learners' speaking has increased. However, it is not a trivial task to detect grammatical errors in oral conversations because of the unavoidable errors of automatic speech recognition systems. To provide corrective feedback, a novel method to detect grammatical errors in speaking performance is proposed. The proposed method consists of two sub-models: the grammaticality-checking model and the error-type classification model. We automatically generate grammatical errors that learners are likely to commit and construct error patterns based on the articulated errors. When a particular speech pattern is recognized, the grammaticality-checking model performs a binary classification based on the similarity between the error patterns and the recognition result using the confidence score. The error-type classification model chooses the error type based on the most similar error pattern and the error frequency extracted from a learner corpus. The grammaticality-checking method largely outperformed the two comparative models by 56.36% and 42.61% in F-score while keeping the false positive rate very low. The error-type classification model exhibited very high performance with a 99.6% accuracy rate. Because high precision and a low false positive rate are important criteria for the language-tutoring setting, the proposed method will be helpful for intelligent computer-assisted language learning systems.

Introduction

Computer-based methods for learning language skills and components are increasingly being used (Stockwell, 2007). These tools assist in the linguistic development of students by providing more language-learning opportunities than human teaching methods. One of the ultimate goals of computer-assisted language learning (CALL) is to provide learners with an environment that facilitates the acquisition of communicative competence, especially oral skills.

As a result, the demand for CALL systems that help language learners develop oral skills has increased, and numerous CALL systems for pronunciation training have been developed to meet this demand (Dalby, 2005; Neri, Cucchiarini, and Strik, 2001). However, pronunciation is only one of the skills required for proficiency in speaking a second language; a learner must also acquire other important aspects of the spoken language, such as morphology and syntax. To account for this fact, CALL systems have been developed to detect grammatical errors in speaking performance, provide learners with corrective feedback, and allow learners to try repeatedly until they manage to produce the correct form.

However, it is not a trivial task to detect grammatical errors in oral conversations because of the unavoidable errors of automatic speech recognition (ASR) systems. The ASR errors make it mostly impossible to employ parser-based methods which have usually been developed to detect grammatical errors in learners' writings (Heift and Schulze, 2007). As grammatical error detection in speaking performance is in a relatively early stage, only a few reports have been published. In addition, most previous studies have lacked proper evaluations to judge the usefulness for language tutoring. In this paper, we propose a novel method capable of handling ASR errors and we provide several evaluation results that are helpful in considering the practicality of the method.

The remainder of this paper is structured as follows. Section 2 briefly describes related studies. Section 3 presents a detailed description of the methods. Section 4 outlines the experimental setup. Section 5 shows the results and discusses their meaning. Finally, Section 6 offers our conclusion.

Related Work

Many research projects have tested the idea of providing pronunciation training using a speech recognizer, but few systems exist that detect grammatical errors in speaking performance and provide learners with corrective feedback.

The Let's Go system (Raux and Eskenazy, 2004) is a spoken dialog system that provides bus schedules. The researchers adapted non-native speakers' speech data and modified the semantic-parsing grammar that originally was developed for the native speaker. Modifications include the addition of new words, new constructs and the relaxation of some syntactic constraints to accept ungrammatical sentences. Based on the recognition result for the user utterance, the system computes its distance to each target sentence using dynamic programming and selects the closest target. If the two match exactly, no correction is produced and the dialogue continues normally. If words were deleted, inserted or substituted by the non-native speaker, they generate both confirmation and correction. The idea is that whenever a non-native speaker utters an ungrammatical utterance, the speaker's goal was actually to utter one of the target sentences, but the speaker made a mistake by inserting, deleting or substituting a word. The evaluation results, however, showed numerous false positives (i.e., the user utterance was judged as ungrammatical although it was grammatical), and most of them were caused by ASR errors. This result clearly shows that we need to take into consideration ASR errors when we judge grammaticality to reduce false positives.

The Spoken Electronic Language Learning (SPELL) system (Morton and Jack, 2005) provides opportunities for learning languages in functional situations such as going to a restaurant or expressing (dis-)likes. Recast feedback is provided if the learner's response is semantically correct but has some grammatical errors. To reduce the confusion between ASR errors and grammatical errors, the system embeds error checking into the speech recognition process. Within the constrained environment defined for SPELL, it is readily possible to predict to a reasonable degree what learners might say at any given stage; similarly, it is then possible to predict certain grammatical errors that they might make. The aim is to develop finite-state network (FSN)-based recognition grammars specifically for non-native speakers that take into account both grammatical and ungrammatical predicted responses (Figure 1). However, this study did not conduct experiments on the performance of the error detection component. Therefore, we implemented the method as the baseline system and performed comparative experiments with our method.

The Development and Integration of Speech technology into COurseware for language learning (DISCO) system (Cucchiarini, Doremalen, and Strik, 2008) is under development and supposed to extend the previous pronunciation training project to morphology and syntax training in well-designed exercises. The aim of the DISCO project is to optimize Dutch learning through interaction in realistic communication situations and provide intelligent feedback on important aspects of speaking. For detecting morphological and syntactic errors, grammatical error simulation software can be used. This software takes appropriate responses as input and expands them to form pools of correct and incorrect responses. Similar to SPELL, the speech recognition module determines which utterance was spoken and the system determines whether errors have been made depending on which of the possible utterances has been recognized. The evaluation on the DISCO system has not yet been performed because the system is currently under development.

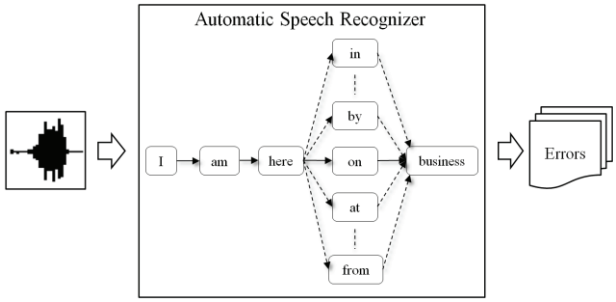

Figure 1: An example of FSN-based recognition grammar to detect possible preposition errors for the correct response "I am here on business"

Grammatical Error Detection

The simplest way to detect grammatical errors in speaking performance while reducing the hindrance of ASR errors is the method employed in SPELL and DISCO. The system takes appropriate responses as input and expands them using a grammatical error simulator to form FSN-based recognition grammars that include both correct and incorrect responses. The system determines whether errors have been made depending on which of the possible utterances has been recognized. However, this approach has severe drawbacks. As the grammar size exponentially increases because of the numerous ungrammatical responses, the recognition performance sharply decreases. Often the recognized hypothesis could be a totally different utterance because the FSN-based Viterbi-decoding searches for the hypothesis at a nearly utterance level. Moreover, even if the recognized hypothesis is similar to the learner's speech, it could be useless for error detection. Because of the grammatical error simulation, we have many similar ungrammatical variants of a correct response. When the learner's utterance is one of the variants, it is highly likely that these similar variants are placed on the N-best hypotheses. However if the right hypothesis is not

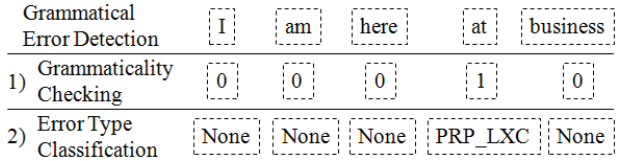

Figure 2: The grammatical error detection model consists of two sub-models

the top hypothesis, the system would produce wrong feedback because the system takes only the top hypothesis. Therefore, we investigate a method that uses ASR systems with an N-gram language model to not get a totally different hypothesis and that considers multiple hypotheses based on confidence scores at a word level by exploiting a confusion network (CN) (Mangu, Brill, and Stolcke, 2000). According to Mangu, Brill, and Stolcke (2000), the posterior probability of a word hypothesis can serve as a confidence score for the word to occur at the position. Unlike previous methods that just rely on the Viterbi-decoding process of ASR systems, this approach allows us to use machine learning techniques that we can try various useful features and have more opportunity to optimize a sophisticated objective function such as a low false positive rate and a high F-score.

Besides ASR errors, there are several factors that make it hard to detect grammatical errors. Because there are far more grammatical words than ungrammatical words in the data, the grammatical error detection model, which is implemented as a classifier, must be constructed to effectively learn from the imbalanced data distribution. When accuracy is the performance measure, using the classifier trained on the highly imbalanced data simply produces the majority class for all test data to achieve the best performance. In addition, the number of error types to classify is relatively large. This can make the model learning and selection procedure vastly complicated. Therefore, to cope with these difficulties, we divide the grammatical error detection model into two sub-models: the grammaticality-checking model and the error-type classification model (Figure 2).

Grammaticality Checking Model

The grammaticality-checking task takes the recognized hypothesis in the form of a CN and determines the grammaticality at each word position in sequence. Even without error type information, the grammaticality-checking function may be very useful for some applications, e.g., categorizing learners' proficiency level and generating implicit corrective feedback such as repetition, elicitation, and recast feedback.

Feature Extraction

To judge the grammaticality, we first extract error patterns from the simulated ungrammatical responses. The error pattern is a 5-tuple consisting of the erroneous word and its two left and two right neighbor words. For example, the error pattern for the proposition error at *'at'* for the utterance *'I am here at business'* will be a tuple <*'am', 'here', 'at', 'business', '-'*[1]>. The error pattern is also tagged with the error type and structural deviation (e.g., deletion or substitution) for the error-type classification task.

When a speech is recognized, at each position in the CN, we extract a feature vector by comparing the error patterns with the segment of the CN, consisting of the target position and the two left and right neighboring positions. We extracted seven features (Table 1) for each error pattern. For example, if the first word in the error pattern exists among the competing word hypotheses at the first position in the CN, then we take the confidence score of the matched word hypothesis as the S1 feature. If there is no matched word hypothesis, we simply set the feature to zero.

The higher the matching scores an error pattern has, the more likely the recognized result has the relevant error in it. Because the number of error patterns is very large and likely uninformative, only the features extracted from top 10 error patterns ranked by the TS feature are used. In addition, we perform a similar feature extraction process at the parts-of-speech (POS) level. We apply POS tagging to both the recognition result and the error patterns to get additional features from the top 10 POS-level error patterns. The POS-level features contribute to raising the recall rate by alleviating the data sparseness problem of lexical-level features. Figure 3 depicts the aforementioned feature extraction process.

Model Selection and Parameter Learning

We use the LIBSVM (Chang and Lin, 2001) Support Vector Machine (SVM) classifier to produce a model that predicts grammaticality. We use a radial basis function (RBF) as the kernel because unlike linear kernels, an RBF kernel allows us to handle nonlinear interactions between attributes (e.g., dependency between the feature SD and the

Feature	Description
S1	Confidence score of the word hypothesis matching the first word in the error pattern
S2	Confidence score for the second word in the error pattern
S3	Confidence score for the third word in the error pattern
S4	Confidence score for the fourth word in the error pattern
S5	Confidence score for the fifth word in the error pattern
TS	Total score of L2, L1, TW, R1 and R.
SD	Indicator of structural error type: 1 for Deletion and 0 for Substitution

Table 1: Description of features extracted from each error pattern to train the grammaticality checking model

[1] '-' is a blank symbol.

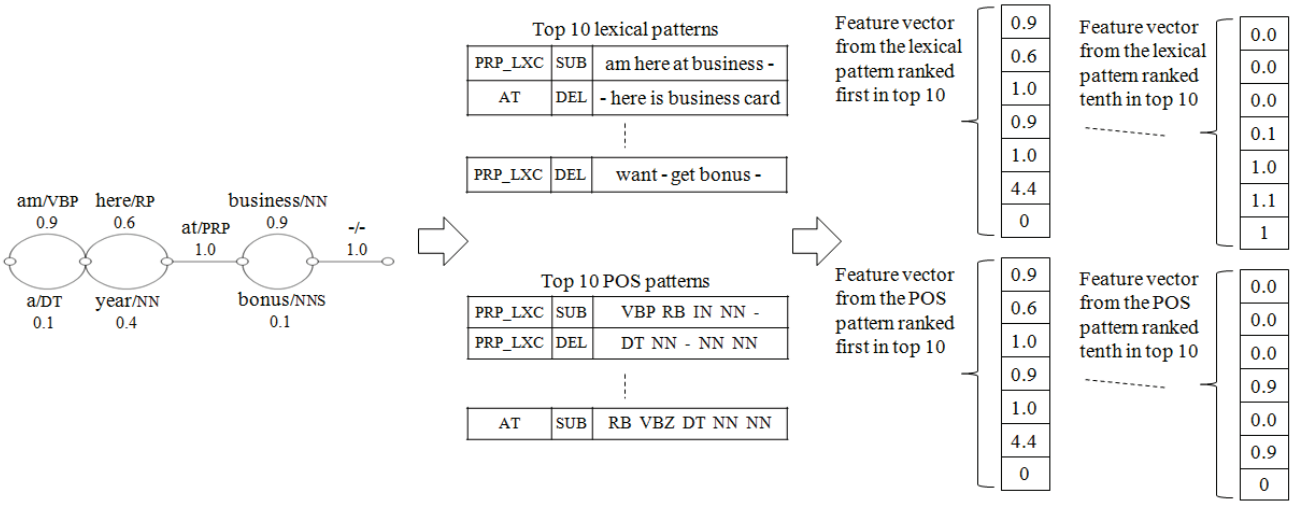

Figure 3: An illustration of feature extraction process. PRP_LXC and AT denote proposition lexical error and article error. SUB and DEL mean substitution and deletion

other features in Table 1) and relationship between class labels and attributes. We conduct simple scaling on the data to avoid attributes in greater numeric ranges dominating those in smaller numeric ranges. We linearly scale each attribute to the range [0, 1]. As mentioned before, the grammaticality-checking task is non-trivial because of the highly imbalanced data distribution. To address this problem, we can oversample the minority class or undersample the majority class to make the data balanced. Also we can implement cost-sensitive learning to assign a larger penalty value to false negatives versus false positives. Without loss of generality, we will assume that the positive class is the minority class (i.e., ungrammatical), and the negative class is the majority class. However, these approaches do not explicitly optimize the objective function that is important for a tutoring setting. For example, false positives (i.e., the user utterance was judged as ungrammatical although it was grammatical) are more detrimental than false negatives. Furthermore, precision is more important than recall. Therefore, in this study, we solve the problem by using a custom objective function instead of using accuracy as a performance measure. The objective function to optimize for this study is:

$$\text{Maximize } F-score$$
$$\text{Subject to } Precision > 0.90,$$
$$False\ positive\ rate < 0.01$$

There are two parameters for an RBF kernel: C and γ. To find the best parameters C and γ that optimize the objective function, we perform a grid-search using 5-fold cross-validation.

Error Type Classification Model

To provide meta-linguistic feedback (i.e., detailed explanations about the grammatical error), we need to identify the error type. Identifying the error type is also beneficial to construct the learner model. Thus, we perform error-type classification for the words that are determined as ungrammatical by the grammaticality-checking model. The simplest way to classify the error type is to choose the error type associated with the top ranked error pattern. But this approach has two flaws: it does not have a principled way to break tied error patterns, and it does not consider the error frequency. Therefore, to solve both problems at the same time, we reorder error patterns by weighting more heavily errors that occur more frequently:

$$Score(e) = TS(e) + \alpha * EF(e),$$

where TS returns the TS feature of the error pattern e and EF returns the error frequency of the relevant error, normalized summing to one. We set the constant α as 0.1 for this study.

Experimental Setup

To evaluate the proposed method, we apply the method to detect grammatical errors of Korean learners of English.

Grammatical Error Simulation

One of the key elements to the development of ASR-based CALL systems for morphology and syntax training is to expand the recognition grammar to include not only grammatical responses but also ungrammatical responses. In SPELL, as each new scenario is developed, it is

essential to create ungrammatical responses by hand. However, using human experts to anticipate various types of grammatical errors and list all possible realizations of the errors is too laborious and costly. Thus, as in DISCO, automatic generation of realistic grammatical errors to create recognition grammars is crucial to the development of such systems. Inspired by Lee et al. (2009), we developed a grammatical error simulator that generates errors that Korean learners of English usually make. To generate realistic errors, expert knowledge of language learners' error characteristics was imported into a statistical modeling system that uses Markov logic (Richardson and Domingos, 2006). A Markov logic network (MLN) can be seen as a first-order knowledge base with weights attached to each of the formulas. A total of 119 MLN formulas were written. For example, English learners often commit pluralization errors with irregular nouns. These errors result because they over-generalize the pluralization rule, e.g., attaching 's/es' to the end of a singular noun, so that they apply the rule even to irregular nouns such as 'mice' and 'feet'. This characteristic is captured by the simple formula:

$$IrregularPluralNoun(s,i) \wedge PosTag(s,i,NNS) \Rightarrow ErrorType(s,i,N\_NUM),$$

where $IrregularPluralNoun(s,i)$ is true if and only if the i-th word of the sentence s is an irregular plural, NNS stands for plural noun, and N_NUM is the abbreviation for noun number error.

We learned the weights of first-order formulas from the NICT JLE corpus (Izumi, Uchimoto, & Isahara, 2005). This is a speech corpus of Japanese speakers learning English[2]. The corpus data were obtained from 1,281 audio-recorded speech samples, 167 of which are error-annotated, from an English oral proficiency interview test. The current version of the error tagset targets morphological, grammatical, and lexical errors and can describe diverse grammatical errors. The error tagset currently includes 46 tags. Because of the space limitation, please refer to Izumi, Uchimoto, and Isahara (2005) for the full list of error types. Lexis errors related to open-word class (i.e., n_lxc, v_lxc, aj_lxc, and av_lxc), were excluded in this experiment because realizing such errors without encountering the data sparseness problem requires a huge amount of learner data. Some other errors (i.e., o_je, o_lxc, o_odr, o_uk, and o_uit) were also excluded because these error categories have not yet been clearly analyzed for practical applications. Error categories that occurred less than five times were also excluded to improve reliability. This results in a total of 23 error types. In addition, we did not explicitly generate insertion errors because many insertion errors appear implicitly as replacement errors in the NICT JLE corpus. The insertion errors, which are not covered in this model, usually relate to vocabularies in open-word classes or are highly unpredictable even when linguistic context is taken into account.

[2] Unfortunately, there is no Korean learners' corpus. But Korean and Japanese speakers learning English have very similar error characteristics because the two languages have very similar grammatical structures.

Data Preparation and Setup of Grammatical Error Detection Models

We took 100 utterances from the NICT JLE corpus and expanded it to form the pool of 5000 utterances using the grammatical error simulator. For the training data, we randomly chose 250 utterances from the utterance pool and ten male Korean speakers to each read 50 utterances, resulting in 500 recordings. For the test data, we randomly chose 50 utterances from the utterance pool and ten male Korean speakers to each read the utterances, resulting in 500 recordings.

We developed our own English ASR system to recognize Korean learners' English more reliably. The acoustic model is based on 3-state left-to-right, context-dependent, 8-mixture, and cross-word tri-phone models, trained on the Korean-Spoken English Corpus (Rhee et al., 2004) using the HTK version 3.4.1 toolkit (Young et al., 2009). A backed-off bigram trained on the 5000 utterances is used as a language model to cover both grammatical and ungrammatical utterances. The lattice output of the speech recognizer is converted to the CN using the lattice-tool (Stolcke, 2002). After constructing the English ASR system, the recognition performance of the ASR system was evaluated on both the traininig and test speech data. The word error rate was 15.20% at the vocabulary size of 530.

For a comparative evaluation on the grammaticality-checking task, we developed the proposed method and also implemented the FSN-based ASR system (FSN in Table 2) which was employed in SPELL and DISCO as the baseline system. In addition, to verify the effect of confidence score-based soft match, we developed an exact pattern match-based method (EPM in Table 2) that judges the recognition result as ungrammatical only when there is an error pattern that exactly matches the sequence formed by picking the word with highest confidence score at each position in the CN. For a comparative evaluation on the error-type classification task, we developed the proposed method with the error frequency estimated from the NICT JLE corpus and implemented the method that takes the error type of the top ranked error pattern as the baseline system.

Model	Precision (%)	Recall (%)	F-score (%)	False Positive Rate (%)
FSN	19.30	18.60	18.94	6.25
EPM	**97.44**	19.64	32.69	**0.04**
Proposed	91.82	**63.82**	**75.30**	0.46

Table 2: Experimental results on the grammaticality-checking task

Results and Discussion

The results showed that the proposed model largely outperformed the baseline FSN model for all metrics (Table 2). It is because the FSN-based Viterbi-decoding exhibited a very low sentence-level recognition performance due to the relatively large size of the recognition grammar consisting of many similar variants for various grammatical errors. This affects not only the precision and recall but also the false-positive rate, where it can be detrimental for language tutoring by frustrating learners. The proposed method also surpasses the EPM model in F-score. It is attributed to the large gain in the recall rate. The proposed method achieves a far higher recall rate than that of the EPM model by exploiting a soft pattern match based on the confidence score. Furthermore, the proposed method lost little precision by virtue of the SVM model optimization to satisfying the constraints on the precision and false positive rate. Both the EPM model and proposed model showed a very low false positive rate. This implies that the proposed method is very suitable for educational applications. For the error-type classification task, the baseline method that does not consider the error frequency showed an accuracy of 95.55%. The proposed method improved the baseline performance by 4.05%. The result of the baseline model is quite good already, but the incorporation of error frequency into the model gives us an additional performance gain.

Conclusion

This study proposed a novel method to detect grammatical errors in speaking performance to provide corrective feedback on grammatical errors. The results showed that for the grammaticality-checking task, the proposed method largely outperformed the two comparative models respectively by 56.36% and 42.61% in F-score while keeping the false positive rate very low. For the error-type classification task, the proposed method exhibited very high performance with a 99.6% accuracy rate. Because high precision and a low false positive rate are important criteria for the language tutoring setting, the proposed method will be helpful for intelligent CALL systems.

Acknowledgement

This work was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2010-0019523).

References

Chang, C., and Lin, C., 2001. *LIBSVM : a library for support vector machines*.

Cucchiarini, C., van Doremalen, J., & Strik, H., 2008. *DISCO: Development and Integration of Speech technology into Courseware for language learning*. Proceedings of Interspeech, Brisbane, Australia.

Dalby, J., and Kewley-Port, D. 2005. *Explicit pronunciation training using automatic speech recognition technology*. In: Yong, Z. (ed.), Research in technology and second language education: developments and directions. Connecticut: Information Age Publishing.

Heift, T., and Schulze, M., 2007. *Errors and Intelligence in CALL. Parsers and Pedagogues*.New York: Routledge.

Izumi, E., Uchimoto, K., and Isahara, H. 2005. *Error annotation for corpus of Japanese Learner English*. Proceedings of the Sixth International Workshop on Linguistically Interpreted Corpora.

Lee, S., and Lee, G. 2009. *Realistic grammar error simulation using Markov Logic.* Proceedings of the ACL-IJCNLP Conference.

Mangu, L., Brill, E., and Stolcke, A. 2000. *Finding consensus in speech recognition: word error minimization and other applications of confusion networks*. Computer Speech and Language, 14:373–400.

Morton, H. and Jack, M., 2005. *Scenario-based spoken interaction with virtual agents.*Computer Assisted Language Learning, 18(3): 171–191.

Neri, A., Cucchiarini, C. and Strik, H. 2001. *Effective feedback on L2 pronunciation in ASR-based CALL*. Proceedings of the workshop on Computer Assisted Language Learning, Artificial Intelligence in Education Conference. San Antonio, Texas.

Raux, A., and Eskenazi, M., 2004. *Using task-oriented spoken dialogue systems for language learning: potential, practical applications and challenges*. InSTIL/ICALL Symposium, Venice, Italy.

Rhee, S., Lee, S., Kang, S., and Lee, Y., 2004. *Design and construction of Korean-Spoken English Corpus (K-SEC)*. In: Proc. ICSLP, Jeju, Korea.

Richardson, M. & Domingos, P., 2006. *Markov logic networks*. Machine Learning, 62(1), 107–136.

Stockwell, G., 2007. *A review of technology choice for teaching language skills and areas in the CALL literature*. ReCALL, 19(02), 105–120.

Stolcke, A. 2002. *SRILM - An Extensible Language Modeling Toolkit*. Proc. Intl. Conf. on Spoken Language Processing, vol. 2, pp. 901-904, Denver.

Young, S. et al., 2009. *The HTK Book.* Microsoft Corporation, Cambridge University Engineering Department.