博士論文（要約）

A Framework for Applying Language Testing Methods to Support Systems of Second Language Use

(言語テスト手法を第二言語使用の支援システムに適用するためのフレームワーク)

江原 遥

Abstract

The combination of globalization and widespread use of text communication over the Internet has led to the growing use of computers for reading and writing texts written in a second language - especially English. Practical application systems for providing automatic support to second language users in reading and writing have been proposed in various fields, including natural language processing (NLP). Moreover, the methodologies used to test second language knowledge have become quite sophisticated over the past several decades in the field of language testing. There is still no framework, however, that enables collaboration between these two fields despite the progress made in each field in recent years. This lack of collaboration means that second language knowledge is not obtainable in practical support systems although it is essential for personalizing support. Personalization of support for second language users is important since their language abilities can greatly vary. Moreover, the mathematical models used for language testing are difficult to apply to practical application systems.

This thesis presents a novel unified framework for practical application systems that enables language testing methods and support systems to be used together. Within language testing, we focus on vocabulary testing because vocabulary knowledge is essential and is used for measuring higher levels of second language knowledge such as reading and writing ability. To determine to which application systems vocabulary testing methods are easily applicable, this thesis categorizes existing application systems supporting second language users by using a proposed *three-handshake* model. With this model, we can see that *reading support* is one of the applications to which vocabulary testing can most easily be applied.

We then identified the properties that language testing models should have to be used in practical application systems. The desired properties are *interpretable parameters using language testing models*, *globally optimal parameter estimation*, *out-of-sample setting*, and *noisy stimuli (word) detection*. Previous models for vocabulary testing, namely the *item response theory* models, do not have all the desired properties. For example, those with noisy word detection are non-convex, and thus lack the first property. We cannot apply language testing models to practical support systems as-is because such models require more robust parameter estimation. To tackle this problem, we developed a novel mathematical framework for deriving convex models with noisy word detection. The derived models have all the desired properties.

Finally, we tested the proposed framework by creating a reading support system that automatically detects and highlights words unfamiliar to users. We also created the first dataset covering a large body of English as a second language user's vocabulary knowledge.

The contributions of this thesis are 1) the proposal of a unified framework that enables language testing to be applied to application systems, 2) the proposal of a mathematical model that satisfies all the desired properties, 3) the demonstration of the proposed framework for reading support, and 4) the creation of the dataset.

Keywords Support Systems, Language Testing, Item Response Theory, Machine Learning

Contents

Chapter 1

Introduction

Natural language processing (NLP) and computational linguistics share the same research community and respectively represent the engineering and scientific aspects of language study using computers. They are categorized as a sub-field of artificial intelligence or more broadly of computer science. We can also categorize computational linguistics as a sub-field of linguistics. Thus, natural language processing and computational linguistics can be viewed as an interdisciplinary study based on both computer science and linguistics.

In this thesis, we focus on the engineering aspect of natural language processing, rather than its scientific aspect. The tasks that could be handled by natural language processing were limited before the emergence and subsequent growing use of the Internet. One critical reason is that the *corpora*, i.e., the set of human-written texts, had been limited in both size and variety. Newspaper articles were one of the few corpora available in the pre-Internet era, so natural language processing research tasks were naturally confined to those regarding such articles.

The emergence of the Internet changed this situation greatly. Texts created by people in general, i.e., the "crowd," suddenly became available. The term *information explosion* was coined to describe the tremendous growth in the volume of texts. Moreover, the variety of texts has also been increasing almost explosively. Because one type of corpora can support many research tasks, the variety of natural language processing research tasks has also increased quite rapidly. The coverage of natural language processing has spread to such a degree that it is not too much to say that it has been redefined.

Not only has the variety of natural language processing tasks increased, some of the new tasks have resulted in great success in both research and industrial areas. Sentiment analysis, for example, can be viewed as such a task. The increased coverage of natural language processing is represented by the task proposed by Pang and Lee (2005). In this task, how much users were satisfied with a product is predicted from the number of user review stars in product review texts.

This task attracted both industrial- and research-minded attention and led to the establishment of such prediction as a sub-field of natural language processing.

We also attempt to expand the range addressed with natural language processing by shedding light on a new line of research. We aim to establish a novel framework for handling each user's second language knowledge in application systems supporting second language users. A second language is a language that a person learns after his or her first, or native, language. Successfully mastering a language is called *acquisition* of a language. For example, English is a second language for most Japanese-speaking people.

As explained later in this chapter, there has been no framework for constructing practical application systems for handling second language users' second language knowledge, although there are frameworks for handling second language knowledge in language testing. We propose a framework that can apply existing frameworks to practical support systems.

1.1 Two separated frameworks for language testing and support systems

This section first introduces the conventional framework for application systems supporting second language users and then discusses and locates the problem with it. Figure 1.1 shows the previous framework. We begin our explanation by defining the concept of **second language knowledge**.

We define second language knowledge as the knowledge one needs to use the language. Note that this definition includes what is not regarded as **knowledge** in the usual sense as a part of second language knowledge: for example, the way the tongue is moved for pronunciation is included in this definition of second language knowledge.

The key difference between a first language and a second language is that many people fail to acquire a second language while virtually no one fails to acquire a first language. Suppose there are two users of a language: one user is fluent in the language while the other is not. It is obvious that any physical differences between them, i.e., differences in the physical structures of their organs, would account for only a small part of the difference. This is because the locations of the muscles around the mouth, tongue, and pharynx are basically the same for all people. Therefore, we can assert that the difference in ability is mainly due to a difference in their mental abilities. We assume that each user has unique second language knowledge and define the difference in their second language knowledge as the mental difference that causes the difference in fluency.

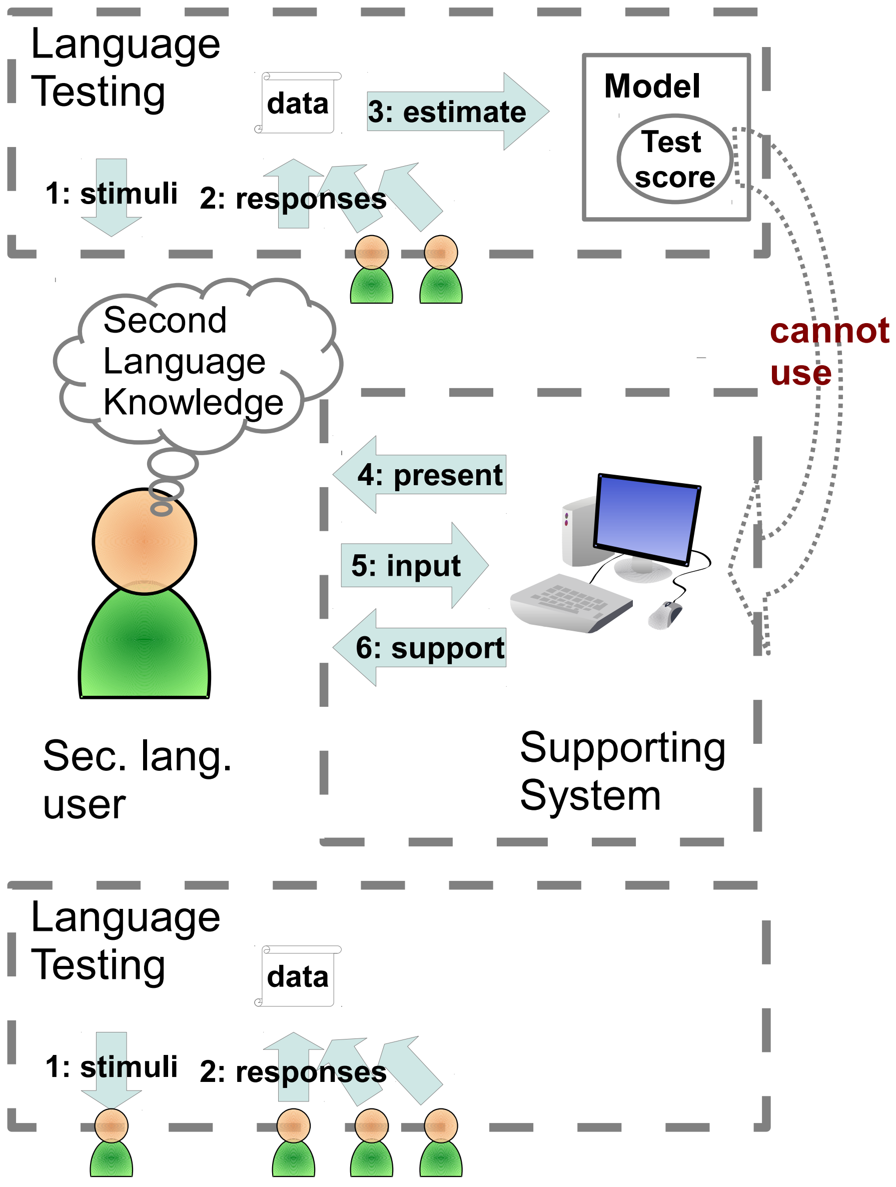


Fig. 1.1. Previous framework

Figure 1.1 shows a second language user and his/her own second language knowledge. The two research fields represented in the figure, language testing methods and support systems, have evolved quite differently, even though they share second language users as their common interest.

Language testing is an established field that uses methodology to more precisely investigate the properties of the users' second language knowledge by using tests. In Figure 1.1, "1: stimuli" represents the language tests administered to second language users, and "2: responses" represents the responses to the stimuli, i.e., the questions or prompts in the language tests. Typical examples of this interaction are tests such as the Test of English for International Communication (TOEIC) and the Test of English as a Foreign Language (TOEFL). The responses of multiple users are aggregated into a data set, and the test scores are commonly calculated from the data set. The calculation of the test scores is represented in Figure 1.1 by “3: estimate” parameters of the corresponding model.

Systems for supporting second language users have similar interaction procedures. The system first presents some material - an introduction telling the user what to do next, for example - to the user. The user then inputs information to the system. Then, on the basis of the information input, the system provides support to the user.

Although the interaction procedure is similar between the two frameworks, few studies have attempted to bridge them. This framework separation has resulted in three problems in particular.

Systems Systems for supporting second language users cannot use the user’s second language knowledge. As a result, for example, it is difficult for the system to personalize its support.

Tests Language tests cannot use the input provided by users to the system.

Users Second language users have to interact with both support systems and language tests, which impose a double burden on them.

Users use support systems and language tests differently. Since support systems benefit the user, they tend to be used for a longer period, with the length of use depending on the system. For example, a typical user might use a support system for about 10 minutes a day for an extended period. In contrast, language tests do not support the users’ use of a second language. They benefit the second language user only through their results. Therefore, they are used for a shorter period because users want to obtain the results sooner. The test time depends on the type of test. More important tests, such as TOEIC and TOEFL, take several hours, which is acceptable to users because these tests can affect job opportunities. Most other tests, like those for research purposes or placement, would not be allowed to take such a long time. They must thus be designed to evaluate the user’s second language knowledge in a much shorter period, say less than an hour.

The contrasting nature of the support system and language test use cases means that unifying their frameworks could compensate for the drawbacks of each. Thus, we propose a unified framework, as shown in Figure 1.2. There are approaches for unifying them that are based on deciding which framework is to be dominant: the first is to devise a support system that can accommodate language test models, and the second is to devise language tests that can accommodate a support system. This thesis takes the first approach.

In the proposed framework, the support system attempts to create models of each language user’s second language knowledge by providing “1: stimuli” and collecting their corresponding

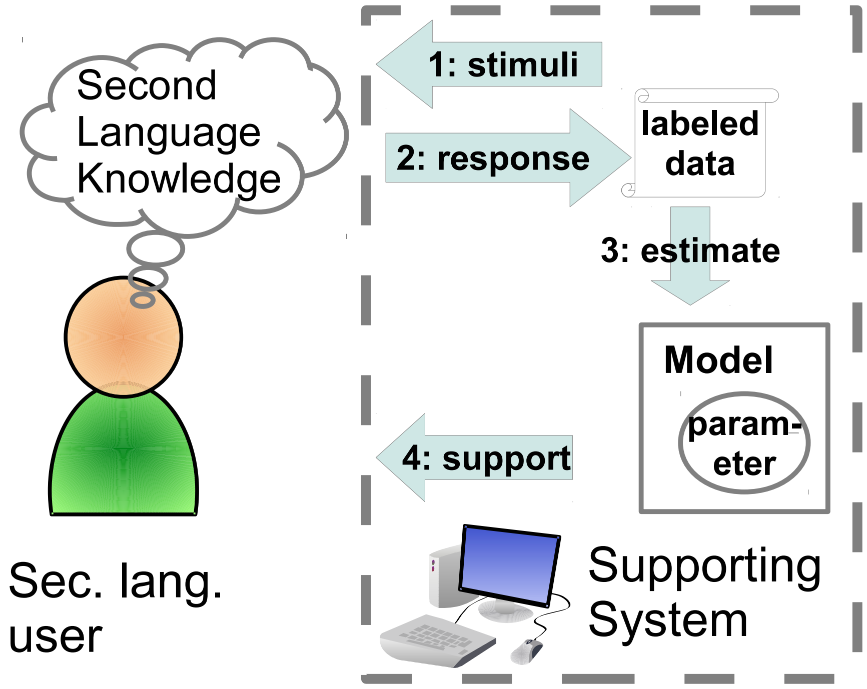


Fig. 1.2. Proposed Framework

"2: response." Note that current support systems are not aimed to model each user's second language knowledge. They simply collect inputs to be processed, and most of them are unaware of each user's second language knowledge: for example, they do not care what words the user may know or the user's academic specialty.

In the proposed framework, however, the support system models each user's second language knowledge by estimating the model parameters from the collected responses, as shown in "3: estimate" of Figure 1.2. Second language knowledge is modeled and summarized as model parameters in this process. For example, if the system is to support the users' use of vocabulary, the system models the user's vocabulary, a component of second language knowledge, and the vocabulary size is stored as a model parameter. Because the system models each user's second language knowledge, there is a personalized model for each user. The system can thus provide personalized support to each user.

The abstraction of the proposed framework is illustrated in Figure 1.3. We call this abstraction the *three-handshake model*. In addition to the interaction between a user and the support system, a person can take the role of the computer support system. The person could be, for example, a language teacher. In the proposed framework, we define three types of *modeling* on the basis of which interaction in Figure 1.2 is to be modeled.

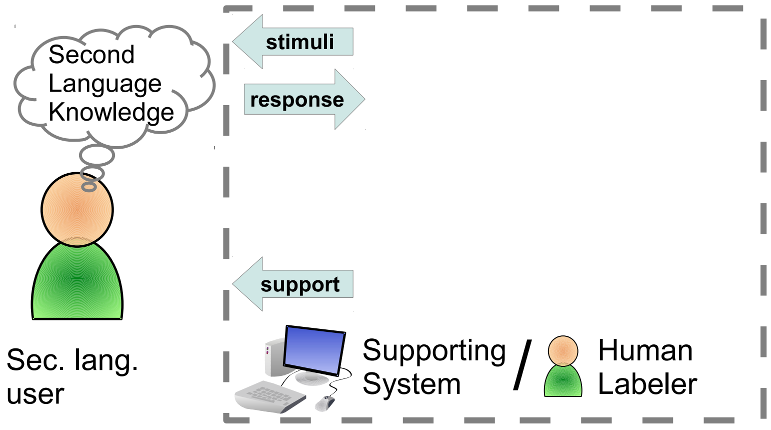


Fig. 1.3. Abstraction of Proposed Framework, or Three-Handshake Model.

Response Modeling models the user's responses from stimuli, support, or both. The user's responses are regarded as labeled data, and stimuli, support or both are regarded as unlabeled data.

Support Modeling models teacher's and/or system's support from stimuli, responses, or both.

Stimuli Modeling models stimuli given to a user from a response, support, or both.

Let us consider the categorization of language testing for example. Table 1.1 shows the categorization of tasks under the proposed framework. The left column shows the target of modeling, i.e., the data that each task is aimed to model. The middle column shows the source of the data used for the modeling. In language testing, a test can be regarded as a stimulus to the user, and response corresponds to the user's responses to the test items. This can be interpreted as language testing attempting to model the user's response from the stimuli. Thus, language testing can be categorized in the first row.

We can likewise categorize tasks that second language user support systems are aimed at as well. To use language testing technology, however, these tasks should be categorized in the same row in Table 1.1. By using this categorization scheme, we can identify the kinds of tasks that are appropriate to be modeled by using language testing. In §2, we categorize previous systems in detail.

1.2 Why vocabulary?

In the previous section, we described our proposed framework for supporting second language users in which language testing, or the modeling of second language knowledge, is unified into a practical support system. To achieve this framework, we need to know the types of second

Table. 1.1. Task categorization under proposed framework.

Target of Modeling (Labeled)	From (Unlabeled)	Example Tasks
response	stimuli	Language Testing, Vocabulary Testing, (part of) Readability Prediction
	support	-
support	stimuli	(part of) Readability Prediction
	response	Grammatical Error Correction
stimuli	response	Automatic Question Generation
	support	-

language knowledge to model. In this section, we describe second language knowledge in detail to identify the types of second language knowledge that are best suited for the proposed framework.

Second language knowledge can be categorized in several ways. A typical categorization is by usage type: knowledge for reading, writing, listening, and speaking. Another typical categorization is by structure level: vocabulary, collocation, phrase, sentence, and discourse knowledge. Note that the categorizations are not mutually exclusive: for example, alphabet knowledge is shared by both knowledge for reading and that for writing.

The categorization by structure level is hierarchical: vocabulary knowledge is used in collocation knowledge, collocation knowledge is used in sentence knowledge, and so on. Therefore, it can be said that vocabulary knowledge is the most basic in this categorization scheme, as it is used in the other levels of knowledge.

Figure 1.4 illustrates the categorization of second language knowledge. The range of each knowledge type is shown in a concentric fashion: the knowledge shown in a larger circle contains the knowledge shown in a smaller circle. For example, sentence-level knowledge includes phrase- and word-level knowledge. That is, sentence-level knowledge, or how to read or write a sentence, contains word-level knowledge, i.e., forms and meanings of words. We can see that word-level second language knowledge, or vocabulary knowledge, is versatile in the sense that it is included in all the upper levels of second language knowledge. Thus, we focused on vocabulary knowledge

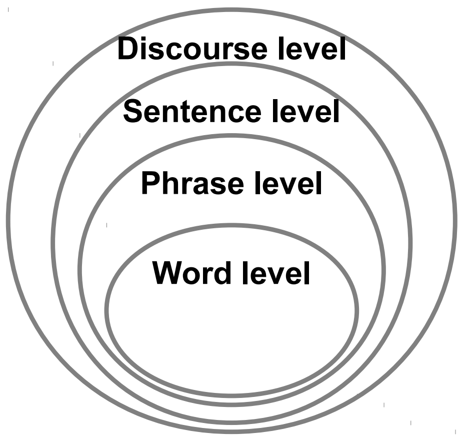


Fig. 1.4. Categorization of second language knowledge.

as the second language knowledge used in the proposed framework.

1.3 Desired properties for models

In the previous sections, we summarized second language knowledge and explained why we focus on vocabulary. This section introduces the *vocabulary prediction* task. The aim of this task is to build a model that predicts, given a word and a user, whether or not the user knows the word. This task conforms to the proposed framework. Using *machine learning* terminology, we can categorize the vocabulary prediction task as a binary classification task: given a word and a user, it predicts whether or not the user knows a word. Therefore, a number of machine learning methods, such as a support vector machine (SVM) for the binary classification task, can be used as predictors. We identified four specific properties that predictors' underlying mathematical models should have.

Interpretable Parameters using Language Testing Models Model parameters should be interpretable under language testing models for applying the language testing methods to support systems. This property restricts the use of models to *response modeling* from stimuli because all language testing models fit in this category as we stated.

Globally Optimal Parameter Estimation Globally optimal parameter estimation is free from local optima and initial-value dependence. This is important for practical support systems because, if globally optimal parameters cannot be estimated, they can easily exhibit poor performance due to noisy data, an inappropriate initial value in parameter estimation, or an unsophisticated estimation algorithm. Globally optimal parameter estimation provides robustness in estimating parameters compared to local optimal parameter estimation.

Out-of-sample setting There are two settings in the vocabulary prediction task for handling new words: *in-matrix* and *out-of-sample*. The in-matrix setting does NOT support new words; i.e., there is at least one training dataset for all the words appearing in the test data. This can be seen as filling in the blanks of a user-word matrix. In contrast, the *out-of-sample* setting supports new words; i.e., some or all words in the test data are missing in the training data. To create the training data, we need to ask users whether or not they know the words. Thus, creation of the training data is financially costly and burdensome for users. In a realistic setting, we can ask users about only a small subset of words then use predictors to predict the rest of them. The out-of-sample setting is more difficult but more realistic than the in-matrix setting.

Noisy stimuli (word) detection Not all stimuli contribute to the response modeling. The user's responses to some stimuli may be too noisy to model or predict. The users' response patterns towards some stimuli may exhibit irregular patterns: for example, the case in which low-ability users respond correctly (positively) and high-ability users respond incorrectly (negatively). More concretely, suppose that normally difficult words (stimuli) are used as the names of well known commercial products and services. In this case, low-ability users may know these words through the product names regardless of the true difficulty of the words. We define the stimuli whose responses are irregular compared to other stimuli as *noisy*. Capturing noisy stimuli can help both support systems and users: for example, if a support system detects noisy stimuli, the system can reduce the burden of users of responding to the stimuli by stopping asking responses towards the stimuli to the subsequent users. The capturing may also help the users in interpreting their own ability: being able to respond correctly to difficult, but noisy, stimuli should not be interpreted as evidence of high ability. Thus, it is desirable for models to somehow detect *noisy* stimuli, although the definition of noise may differ from model to model. Note that stimuli in this thesis correspond to words when we focus on vocabulary tasks.

Although other classifiers, such as SVMs, can be used for prediction as well, they are out of the scope of this thesis because they cannot be interpreted within the scope of previous language testing models, namely the *item response theory* (IRT) models. Thus, throughout this thesis, we focus on the IRT models. Table 1.2 summarizes the IRT models explained in this thesis: the Rasch model, or one-parameter IRT model, two-parameter IRT model, and three-parameter IRT model. The Rasch model is the simplest, and the other models are extensions of the Rasch model. In

Table. 1.2. Desirable properties of models.

	Interpretable	Global optima	Out-of-sample	Noisy Stimuli
Rasch (One-parameter IRT)	✓	✓	-	-
Two-parameter IRT	✓	-	-	✓
Three-parameter IRT	✓	-	-	✓
Shared difficulty	✓	✓	✓	-
Proposed	✓	✓	✓	✓

addition to the proposed framework, this thesis also proposes a mathematical model that defines a predictor in §5. In Table 1.2, we can see that only the proposed model has all the desired properties.

1.4 Contributions

This section summarizes the contribution of this thesis and explains what was achieved towards our goal.

Unified framework This thesis is the first to propose a framework that applies language testing methods to practical support systems to support second language users.

Models Previous models for language testing are difficult to apply to practical support systems as-is. We listed above the four properties that mathematical models for practical support systems should have. We propose a convex model that supports all four properties. We further generalize the convex model to derive other convex models.

Demonstration of proposed framework To demonstrate the effectiveness of the proposed framework, we present a concrete and practical example of using the framework for reading support.

Creation of data set Our dataset of second language users' vocabulary knowledge is the first to almost comprehensively cover English vocabulary.

The structure of this thesis is as follows. In §2, we survey and categorize previous studies on systems that support second language users to identify to which applications the language testing models are easily applied. In §3, we survey previous vocabulary test studies to show that previous language test studies were NOT designed for integration into support systems.

In §4, we introduce previous statistical models used in language testing, namely the item response theory models. We explain the assumptions underlying the IRT models and that these models cannot meet all the desired properties due to meeting these assumptions. We explain the properties that the previous models satisfy among the desired properties. Then, in §5, we present our proposed convex model and explain that it has all the desired properties due to relaxing one of the assumptions underlying the IRT models. In §6, as a concrete example of the proposed framework, we also present our original system and demonstrate its effectiveness and discuss the creation of the dataset.

Finally, in §7, we conclude with a summary of the key points and a description of future work.

The three chapters, §2, §3, and §4 are designed to be able to be read as surveys of previous work in each field. Our original work is concentrated in §5 and §6. Previous ideas involving §5 and §6 are discussed in the survey-like chapters, §2, §3, and §4.

Chapter 2

Systems to Support Second Language Users

This chapter surveys existing systems to support second language users. The tasks these systems have to deal with are numerous, and the goal of this chapter is to determine which tasks fit in the proposed framework by surveying tasks and studies in which systems support second language users.

The major tasks related to support for second language users in NLP can be compiled into three basic categories:

Grammatical Error Correction This task automatically corrects grammatical errors made by second language users when writing second language text. It is mainly beneficial to second language users rather than second language teachers.

Readability Prediction This task, when given second language texts, predicts the difficulty of the text so that second language teachers can easily judge whether the level is appropriate for their students. It is mainly beneficial to second language teachers rather than second language users.

Automatic Question Generation This task, when given second language texts, automatically generates tests that users can take for practice.

We use Table 1.1 to categorize tasks in this chapter.

2.1 Grammatical Error Correction

Grammatical error correction is a task to automatically correct the "grammatical" errors of second language users. Although the name of the field includes "grammatical", modern grammatical error correction research does not confine itself to grammatical errors: simple spelling errors and the selection of appropriate words are also studied.

In the categorization by the proposed framework, grammatical error correction can be viewed as a *support modeling from response*. In grammatical error correction, *response* corresponds to essays that users write and *support* corresponds to the same essays rewritten by native speakers or by language teachers. As grammatical error correction tries to model the rewritten essays from the raw essays, it can be viewed as a support modeling from response.

The role of stimuli in grammatical error correction can be viewed in several ways. One simple view is that there is no stimuli in grammatical error correction. Another view is that, if some texts that designate a topic on which the users should write are provided to the users, designating the texts can be regarded as stimuli in grammatical error correction.

A seminal paper in the grammatical error correction field is (Knight and Chander, 1994). It focuses on the selection of articles ("a", "an", and "the") using $200,000$ automatically collected rules. Although rule-based, this paper still contains elements of modern grammatical error correction studies: it narrows the problem of the grammatical error correction task down to a *classification problem* of *targets of correction* within a sentence. In fact, the following studies can be interpreted as having evolved in two ways: first, by extending the *classifier* to make it more sophisticated, and second, by extending the targets of corrections to other areas, e.g., prepositions.

For the first point, we can categorize previous studies by classifiers used, as follows:

Decision trees Nagata et al. (2006)

Log-linear model Han et al. (2006)

Semi-supervised Gamon (2010)

Comparison of many types of classifiers Rozovskaya and Roth (2011)

Graph-based Bao et al. (2011)

For the second point, we can categorize previous studies on grammatical error correction by their targets of correction, as follows:

Articles Minnen et al. (2000); Knight and Chander (1994); Lee (2004); Nagata et al. (2006); Han et al. (2006)

Prepositions Baldwin et al. (2009); Rozovskaya and Roth (2011)

Articles and prepositions Dahlmeier and Ng (2011a), Tetreault and Chodorow (2008)

Verb selection Lee and Seneff (2008); Liu et al. (2011)

Verb tense and aspect Tajiri et al. (2012)

Spelling Park and Levy (2011)

As grammatical error correction is not categorized into a modeling of response, on which language testing methods are based, it is difficult to utilize language testing methods directly for grammatical error correction.

However, some grammatical error correction can perform a very early personalization. One of the simplest forms of personalization is to make use of information from the users' native language. Nagata et al. (2008) reported that English texts for Japanese users frequently include romanized Japanese words and proposed methods to detect Japanese words mixed in texts written by Japanese users of English as a second language. Dahlmeier and Ng (2011b) targeted Chinese English users and showed that grammatical error correction can be improved by taking semantic similarity in the users' native language, i.e., Chinese, into account.

Regarding second language users' native language, some studies try to support the writings of these users by allowing language-mixed input and translating. For example, Ma et al. (2008), an early work of this approach, take English text with some Japanese mixed as input, translate the Japanese parts, and output fully English texts. Some recent studies have made this process more interactive: rather than taking native language-mixed texts as input, they translate native language texts while inputting native languages. Huang et al. (2012b,a) proposed Web-based systems using this approach for Chinese English users. More recently, Hayashibe et al. (2012) proposed a similar system for Japanese users of English as a second language.

Ultimately, it is difficult to personalize grammatical error correction by measuring second language knowledge using language testing technology through the interaction between the users and systems because grammatical error correction is not categorized into the response modeling from stimuli, on which language testing stands. To our knowledge, no research has applied users' predicted vocabulary to grammatical error correction.

2.2 Readability Prediction

Readability prediction is a task to predict how difficult a document is for users, mainly to support language teachers. Seminal and classic papers for this task came long before those on grammatical error correction (Flesch, 1948; Dale and Chall, 1948). Therefore, related works on readability prediction are too numerous to cover in this thesis. Because readability prediction itself is not the main focus of this thesis, in this section we briefly survey previous studies of readability prediction that partly follow (François and Fairon, 2012).

It is notable that readability prediction is beneficial mainly to language teachers rather than language users. It mainly aims to help language teachers find second language texts appropriate for their students. Therefore, difficulty gold standard labels created by language teachers are usually used to evaluate a system's performance rather than difficulty labels created by users.

Categorizing readability prediction in the proposed framework in Table 1.1 is somewhat complex. If readability prediction poses a document to a user directly, and the user responds with how well he or she can read the texts directly to the system, the task is obviously categorized into a response modeling from stimuli, i.e., the document given to users corresponds to the stimuli, and users' responses are as-is responses. However, because many readability tasks aim to support language teachers, not users, this is not the case in many readability prediction tasks. Instead of collecting responses from users, most readability prediction studies typically assume that language teachers label texts as to whether or not they are appropriate to use in their classes. Most readability prediction tasks aim to predict, from a document, the labeling that language teachers attach to documents. If we view this labeling of texts by the teacher as a part of *support*, we can categorize this task as support modeling from stimuli, where the support corresponds to the teachers' labeling and the stimuli correspond to the given documents. This is why there are multiple "readability predictions" in Table 1.1.

Many readability prediction studies have extensively investigated the types of features that are beneficial to prediction from texts. Recent studies also focus on which classification method is beneficial to readability prediction.

Si and Callan (2001) proposed a measure of reading difficulty of Web pages, unigram language with surface features, more accurate than Flesch-Kincaid formula (Si and Callan, 2001). Collins-Thompson and Callan (2005) used "nai:ve Bayes classification that combines multiple language models". Heilman et al. (2008) used combination of lexical features and grammatical features. They also compared statistical models of nominal, ordinal,and interval scales of measurement. They found ordinal regression was the most effective. Schwarm and Ostendorf (2005) used SVM with feature combination from traditional reading level measures, statistical language models, and other language processing tools. 吉見 毅彦 et al. (2008) tried to predict reading time using machine learning, namely SVM-based model. Tanaka-Ishii et al. (2010) proposed readability by using "learning to rank" in machine learning. Specifically, they used SVM-rank, a pairwise ranking approach.

To summarize this section, as far as we surveyed, we could not find a readability prediction studies that can be categorized into a response modeling from stimuli in Table 1.1. This means that it is difficult to apply language testing technology in current setting. However, the nature of readability prediction task is suitable for application of language testing technology. The major obstacle seems to be the information about how well users could read the documents. This information should be measured by using methods such as readability comprehension tests, which usually should be made by hand. If this information is easy to collect, language testing technology is applicable to reading prediction task as well, and it is able to create a system that can predict, given a document, how well a user can read the document by using language testing technology.

2.3 CALL and UI Systems

In the previous section, we saw that users' language background information has not received much focus in readability prediction studies because, if texts are printed onto papers, there is no means of obtaining the users' language background information. In contrast, studies on computer-aided language learning (CALL) put much emphasis on users' responses to investigate how well the computers aid the language learning.

In the CALL literature, providing a definition, or a *gloss*, from a dictionary for words and phrases in sentences and documents displayed on a computer screen is usually called *glossing*. With the rise of pervasive use of computers, a substantial amount of research questions has been posed in 1990s and 2000s to investigate how this *glossing* can help second language users.

There are some notable research questions in this branch of the studies:

1. Can glossing help reading comprehension?
2. Can glossing help vocabulary learning?
3. Types of glossing: text, pictures, movies (multimedia), and combinations.

As for the first issue, most studies seem to agree that glossing helps with reading comprehension. Yanguas (2009) writes that "glosses have been shown to be of help to the student in the comprehension of a written text" by citing many articles such as (Davis, 1989; Lomicka, 1998).

As for the second issue, studies seem to agree that glossing can help vocabulary learning. For example, Hulstijn et al. (1996) reports a positive result. However, careful experimental settings seem to be required to obtain a significant improvement. For example, Yanguas (2009) reports that they could not obtain a significant increase of the vocabulary size.

The third issue is to investigate which types of glossing are most beneficial to second language

NOT

PUBLIC

Fig. 2.1. Example of a picture gloss in the Appendix, (Yanguas, 2009).

users. The users' recall of vocabulary and comprehension of reading texts are of particular focus. On this issue, many studies conclude that glossing is more beneficial than a paper dictionary in several points. For example, Yanguas (2009) compared *text gloss*, shown in Figure 2.2, *picture gloss*, shown in Figure 2.1, and *combination gloss*, shown Figure 2.3, to see how users reacted to each. As is the case with these studies, a second language text to read is provided to the subjects and the subjects' response in checked by a *pre-post* test, i.e., a test that the subjects are supposed to take before and after reading the text. With the second language text, the *target vocabulary*, the words that are used in the text and target of investigation are set.

To determine how the users notice and recognize words, a methodology called *think-aloud* is used in this field. The subjects are forced to speak what they are thinking about during the experiment of reading texts. Yanguas (2009) also employs this methodology.

Yanguas (2009) found that "all multimedia gloss groups noticed and recognized significantly more of the target words than the control group". They also found that "no significant differences were found among any of the groups in production of the target vocabulary items". Finally, they

Fig. 2.2. Example of a text gloss in the Appendix, (Yanguas, 2009).

found that, in reading comprehension, “the combination gloss group significantly outperformed all other groups”.

In summary, it can be said that glossing is helpful for increasing vocabulary size and that it is even more helpful for improving reading comprehension.

2.3.1 Practical glossing systems

In CALL studies, the effectiveness of systems is usually far more important than the availability of systems. However, practical systems need to be easily available to second language users. To this end, some practical studies and systems have placed emphasis on implementing glossing systems with a focus on Web-based availability.

Glossing systems for Web pages have been implemented in two forms: as desktop applications that retrieve Web pages and perform the glossing on the user’s local machine and as Web applications in which Web page retrieval and glossing are performed on a Web proxy server instead of

NOT

PUBLIC

Fig. 2.3. Example of a combination gloss in the Appendix, (Yanguas, 2009).

on the user's local machine. While desktop glossing systems are fast because they do not require communication between the Web proxy server and the user's local machine, they are not suitable for accumulating user click logs because the logs are distributed across many local machines in desktop applications. As far as we surveyed, most glossing systems are Web-based.

The systems we introduce in this section are:

rikai.com (T. D. Rudick, 2001) **rikai.com** is an early Web-based glossing system created by a programmer, not an academic expert. It has been running since 2001. A screen shot of this system is shown in Figure 2.4. Its main characteristic is that it can gloss any Web page if the URL of the Web page is specified. In Figure 2.4, the mouse cursor is over the word "warrant", and thus this word is glossed with its meaning in Japanese. The system supports English-to-Japanese glossing and Japanese-to-English glossing.

popjisyo Coolest.com Inc. (2002) **popjisyo** is another glossing system that, while developed using similar technology to **rikai.com**, is unique in that it supports glossing between many

NOT

PUBLIC

Fig. 2.4. A screen shot from **rikai.com**.

languages, including Japanese to English, Japanese to German, English to Japanese, English to Korean, English to Spanish, Chinese to Korean, Chinese to English, and Korean to English. Figure 2.5 shows an example of glossing by using popjisyo. By the putting the cursor over a word, it pops up its meaning.

Reading Tutor (Kawamura and Kitamura, 1997) Reading Tutor is a Web-based glossing system created for academic purpose. Given a Japanese sentence, it can gloss the sentence with English, German, Dutch, Russian, or Spanish. Unlike **rikai.com**, it cannot accept Japanese Web pages as the input. Figure 2.6 shows the top page of Reading Tutor. Users can input the sentences to gloss and select the desired language. Users can also access a

NOT

PUBLIC

Fig. 2.5. Example of a text gloss in pop-jisyo.

pre-determined difficulty level of the vocabulary used in the sentences. Figure 2.7 shows a glossing by using Reading Tutor. The input sentence is shown on the left-hand side of the screen. If the user wants to know the meaning of a word shown in the sentence, he or she can gloss, or reveal its meaning, by clicking the word. The gloss is shown on the right-hand side of the screen. Users can also use this system to create their own wordbooks.

Asunaro (Abekawa et al., 2002) Asunaro is a system for Japanese users created by academic experts. Like the Reading Tutor system, this system cannot gloss a Japanese Web page and rather focuses on enabling deeper analysis of the input text. This system supports glossing from Japanese to English, Chinese, Thai, Malay, and Indonesian because Japanese users of these languages are many. Figure 2.8 shows a screen shot of the Asunaro system. The screen is divided into three frames: the left, the right, and the bottom. In the bottom frame, an input text is shown. Like the Reading Tutor system, users of the Asunaro system can reveal the meaning of a word by clicking the word appearing in the bottom frame. Its meaning is then shown in the right frame. Unlike other systems, Asunaro can show a dependency parse tree of the input text by using CaboCha, a Japanese dependency parser

NOT

PUBLIC

Fig. 2.6. Top page of Reading Tutor.

(Kudo and Matsumoto, 2002). Figure 2.9 shows a screen shot in which Asunaro shows the result of a dependency parse tree. The screen in Figure 2.9 is divided into four frames showing the raw output of CaboCha (upper-left), its parse tree (lower-left), and the parse tree in a representation using boxes (upper-right).

Zurukko (Nobori, 2010) Zurukko is a glossing system that glosses English Web pages with Japanese and Chinese. This system was developed by the famous super programmer Daiyu Nobori. The system's concept is similar to the system that we will explain in §6 except that this system does NOT use an automated predictor to predict whether the user knows a word in the Web page or not. Zurukko can record the words that a user knows and glosses only

NOT

PUBLIC

Fig. 2.7. A screen shot of glossing in Reading Tutor.

the words that he or she has not yet memorized. For example, Figure 2.10 shows the top page of the Zurukko system. The number of words (in types) that the user has reported as known words are shown within the red rectangle. Figure 2.11 shows the glossing of a Web news page from *cnn.com* by using the Zurukko system. If the user clicks a word on this screen, Zurukko records that he or she has memorized the word and hides the gloss. If the user clicks the word again, Zurukko cancels the record that the user has memorized the word and the word then appears again. While all words on the Web page are glossed with Japanese, we can see that, within the red rectangle, there are some words that the system did not gloss. These are the words that the user reported he/she had memorized.

NOT

PUBLIC

Fig. 2.8. A glossing in the Asunaro system.

Desktop glossing systems are usually implemented as add-ons for Web browsers, such as **Fire-Dictionary** Nori (2005) and **popIn** (popIn Inc., 2008). They work as add-ons for the **Firefox** Web browser, as well, and Google Toolbar works as an add-on for Firefox and Internet Explorer.

However, as far as we know, there is no application that can estimate a user' s language ability or predict words unknown to him or her from the accumulated click logs, though some applications might simply accumulate them for quick access to the words previously clicked.

Fig. 2.9. A screen shot from the Asunaro system. It displays the dependency structure of the inputted text.

2.4 CAVOCA System

The glossing systems that we have surveyed so far aim to support reading and *incidental learning* of vocabulary during the reading. Here, incidental learning refers to the phenomenon of an individual learning words while reading or listening. In contrast, the phenomenon of an individual intentionally trying to learn words by using, for example, a wordbook is called *intentional learning*. The relation between incidental learning and intentional learning is well studied in (Hulstijn, 2001). As the users of glossing systems are supposed to learn vocabulary incidentally during

NOT

PUBLIC

Fig. 2.10. Top page of the Zurukko system. The red rectangle is drawn by us.

reading, vocabulary learning is not always the focus of the systems.

There have been a few systems proposed to support the intentional learning of vocabulary. That is, rather than support users to learn vocabulary indirectly through actions such as reading, these systems are designed to aid users in learning vocabulary directly by using computers. This section introduces a selection of such studies.

A seminal paper in this field is (de Groot and Keijzer, 2000). They developed a vocabulary learning system in order to see how second language words are learned by users. They found that "cognate and concrete words were easier to learn and less susceptible to forgetting than noncognates and abstract words" and that "word frequency hardly affected performance."

NOT

PUBLIC

Fig. 2.11. Glossing in the Zurukko system. The red rectangle is drawn by us.

A more practical system in this field is **iKnow** (once known as **smart.fm**). The iKnow system helps users in memorizing words and phrases. Figure 2.12 shows a screen shot from the iKnow system. A user can choose a *course* which consists of a set of words and phrases to learn. While most courses are provided by the system developer, users can register their own words and phrases to a course. Figure 2.13 shows another screen shot from iKnow. Users can check the current status of their progression through a course. Once they choose to start the course by making the selection on this screen, the system proceeds to Figure 2.14. An English word and a Japanese gloss are shown, and users are tested on them afterwards.

NOT

PUBLIC

Fig. 2.12. A screen shot from iKnow. Users can choose *courses*.

2.5 Chapter Summary

In this chapter, we surveyed studies from a variety of areas to investigate how vocabulary predictions are used in practical applications.

In natural language processing, *grammatical error correction* and *readability prediction* have been the two biggest topics. In grammatical error correction, although early studies completely ignored users' second language knowledge, some recent studies have handled personalization. In terms of readability prediction, few studies have focused on users' second language knowledge.

NOT

PUBLIC

Fig. 2.13. A screen shot from iKnow. The current status of course progression is shown.

This is mainly because users have no means to effectively provide readability prediction systems with their second language knowledge when texts are provided as printed papers.

In contrast, in the computer-aided-language learning (CALL) field, users have means to indicate their second language knowledge to systems through interaction. In this field, how these interactions can be beneficial to human language learning is of key focus. Regarding this thesis, the effect of glossing, or the display of word meanings in text or multimedia form, has been extensively investigated. Many studies have concluded that glossing is as good as or more beneficial to the vocabulary recall and comprehension of reading texts than paper dictionaries. These studies demonstrate the effectiveness of glossing, which our system (proposed in §6) also uses.

NOT

PUBLIC

Fig. 2.14. A screen shot from iKnow. Users can learn words in a course word by word.

Chapter 3

Vocabulary Studies and Vocabulary Prediction

In the previous chapter, we surveyed engineering applications to see how they relate to the vocabulary prediction task, which we defined and introduced in §1. In this chapter, we survey existing second language vocabulary studies and how they relate to the vocabulary prediction task. In the categorization of the proposed model, Table 1.1, all studies in this section can be categorized into a response modeling from stimuli.

Vocabulary studies can be divided into those targeting first language and those targeting second languages. Research communities are also divided by whether their target is the first or second language. Vocabulary of the first language is the focus of, for example, psycholinguistics researchers. Their aim is mainly to investigate human reaction towards first languages and the acquisition of the first language. The learning of second languages is less relevant in first language studies.

Second language vocabulary is, in contrast, extensively studied to investigate methods to ease the burden of second language acquisition in research fields such as applied linguistics, language testing, and second language acquisition (SLA), especially Teaching English as a Second Language (TESOL) if English is the target for the second language. Vocabulary assessment studies of second languages is a sub-field of these areas.

One key property of vocabulary lies in its size: obviously, second language vocabulary is too large to test all of the words in a second language exhaustively. Therefore, methods of vocabulary assessment, or measurement, is one of the main focuses in these studies. To this end, a number of assessment tests has been proposed.

As for vocabulary assessment tests, however, researchers seem to agree that the purpose of tests - or the target of the measurements, to state it more precisely - must be set in the first place and

Table. 3.1. Purpose of tests

Name of test	Purpose of test	Quoted from
Vocabulary Size Test	"The Vocabulary Size Test is designed to measure both first language and second language users' written receptive vocabulary size in English."	p.1, Nation (2012b)
EFL Vocabulary Tests	"The tests in this book provide a quick method of profiling the vocabularies of users."	p.5, Meara (2010)

no single test measures vocabulary perfectly. In fact, clear statements are usually found in the instructions of tests. We summarize the purposes of the tests that we introduce in this chapter in Table 3.1.

In Table 3.1, we can see that the Vocabulary Size Test cautiously claims that it is a measure of "written receptive vocabulary" of the test-takers. This is a noteworthy mention in that it makes a distinction between **receptive** and **productive** and between **written** and **auditory**. The distinction can be simply summarized as follows:

Receptive Vocabulary that a user can understand when the word is used in a given written or auditory stimuli.

Productive Vocabulary that a user can utilize when writing and speaking.

Many studies show that the size of **productive vocabulary** is smaller than that of **receptive vocabulary**. Simply stated, we cannot use some words in written composition or in conversation even though we can understand their meaning when they are used in academic papers. As the size of receptive vocabulary and productive vocabulary differs, they must be measured separately. Also, it is important to make a distinction on receptive vocabulary between written and auditory. For example, as many Japanese speakers of English cannot distinguish "r" from "l", some pronunciation-related mistakes are limited to only auditory tests.

Still, of the various kinds of vocabularies, measuring written receptive vocabulary is practically the easiest. In the measurement of auditory receptive vocabulary, we need to carefully design, for example, the pronunciation of which region should be adopted since the pronunciation of a language usually differs by region. Throughout this thesis, we solely handle written receptive vocabulary.

3.1 Read and Chapelle's Framework for Vocabulary Assessment

While we are proposing a unified framework for practical support systems, in the field of vocabulary testing, a framework for vocabulary assessment was proposed in (Read and Chapelle, 2001). Although it should be noted that the purpose of the framework is tests, and our purpose is practical application systems, we introduce this work because it is useful to grasp how frameworks are built in vocabulary assessment. In this framework, they discuss three important features of vocabulary assessment, which they call the three "dimensions" of vocabulary assessment, or vocabulary tests. As we regard this characterization of vocabulary tests as an effective way to summarize and compare various vocabulary tests, we introduce their framework in this section. Figure 3.1 shows the characterization that they made on vocabulary tests. Each dimension is dichotomous and comes with its own definitions.

The first dimension is **discrete** and **embedded**. In the definition of discrete and embedded, the word "construct" is used. Here, a construct is, roughly speaking, a target that an assessment test tries to measure. Thus, a discrete test, defined as "a measure of vocabulary knowledge or use as an independent construct", is a test that tries to measure the test-takers' vocabulary knowledge or use itself. For example, the Vocabulary Size Test Nation and Beglar (2007), shown in Figure 3.2, is a discrete test. The test's details are explained in §3.2, but briefly, the Vocabulary Size Test is a test that is designed to measure vocabulary size. As vocabulary size is obviously a measure of vocabulary knowledge, the Vocabulary Size Test can be regarded as a discrete test.

In contrast, an embedded test is not designed to measure the vocabulary knowledge itself. An embedded test is defined as "a measure of vocabulary of which forms part of some other, larger construct" in Figure 3.1. The phrase "some other, large construct" means that vocabulary knowledge itself is NOT the target of the measurement of the test. Rather, the test tries to measure something other, or larger, such as reading ability. For example, Figure 3.1 lists "TOEFL vocabulary items" as an embedded test because its target of measurement is NOT vocabulary size but rather the reading ability of users, and vocabulary items, or problems, are only used to measure the reading ability.

The second dimension is **selective** and **comprehensive**. Note that this distinction is whether being selective or comprehensive NOT to the vocabulary knowledge of the test-takers but to the

NOT

PUBLIC

Fig. 3.1. Three dimensions of vocabulary assessment. This figure is taken from Figure 1 of (Read and Chapelle, 2001).

vocabulary in the stimuli or response. For example, suppose a second language document is provided to the test-takers prior to a test for a reading task, prior to the vocabulary test. Then, if all the words in the document are asked in the vocabulary test after the reading task , the vocabulary test is said to be comprehensive. If some of the words in the document are asked in the vocabulary test after the reading task, the vocabulary test is said to be selective. If there is no such prior reading task, usually, a test can be regarded as being selective.

The third dimension is **context-independent** and **context-dependent**. This dimension is relatively easy to understand. If the test-taker can choose the correct answer simply by the stimulus word, the test can be regarded as context-independent. Otherwise, i.e., if the test-taker cannot choose the correct answer simply by the stimulus word, and he/she needs to take other information, or the context of the stimulus word, into account, the text is said to be context-dependent.

With all the dimensions, it can be said that the dependence on context determines a large part of the characteristics of a test: if there is no prior reading task or listening task, the test is by definition **context-independent**, **selective** because we cannot determine how large the test should be to make the test comprehensive without context, and **discrete** because there is no construct involved other than the vocabulary knowledge.

In this thesis, we propose a framework of support systems that can measure vocabulary knowledge within the interaction. To specify a context inevitably leads to specifying a concrete implementation of the support system framework. To make the framework general, we need to avoid specifying concrete contexts.

Thus, throughout this thesis, we mainly deal with discrete, selective, and context-independent vocabulary tests. All the tests that we introduce in this chapter share these features in common as well.

3.2 Vocabulary Levels Test and Vocabulary Size Test

The Vocabulary Levels Test Nation and Waring (1997) and its successor, the Vocabulary Size Test Nation and Beglar (2007), is one of the most famous vocabulary tests to measure vocabulary size. These tests are multiple-choice format and, in the characterization of Read and Chapelle (2001), are designed to be discrete, selective, and context-independent. Figure 3.2 shows an excerpt of the test.

For readers of this thesis, the Vocabulary Size Test is probably easy to understand. The test is a set of items and each item contains a word that the item tries to measure. The target word is embedded within a **non-defining** sentence. After the sentence, four choices, one of which is correct, follow.

Although the word is placed in a sentence, or a context, this context is set to be **non-defining**. That is, the item is not semantically ambiguous and is designed to be solvable without consulting its context. For example, the word "bank" is polysemous and has two meanings: one is embankment, and the other is financial agency. However, the test-takers will not be required to distinguish these two meanings from the sentence, or context.

An interesting point of the Vocabulary Levels Test and the Vocabulary Size Test lies in the selection of words to be tested. This selection is elaborately designed from the definition of

NOT

PUBLIC

Fig. 3.2. Example of Vocabulary Size Test. This page is taken from (Nation, 2007).

"word". The size of vocabulary differs largely by the definition of word. For example, typically, whether conjugated forms are counted or not makes the count differ greatly: "study", "studying", and "studied" are counted as 3 if conjugated forms are counted and as 1 if conjugated forms are ignored.

The Vocabulary Levels Test and the Vocabulary Size Test propose a "word family" as a unit of counting words. The definition of word family ignores conjugated forms: i.e., all "study", "studying", and "studied" are unified into the word family of "study" when using word families as a counting unit. The counting using word family also ignores polysemous words: e.g., using the aforementioned example, "bank" as an embankment and "bank" as a financial agency are not distinguished in "word family" counting. In the counting using word families, a test-taker knows the word "bank" if the test-taker knows at least one of the two meanings. Polysemous words are designed to be avoided in the tests because handling them complicates the interpretation of the tests.

As for the vocabulary size, a relatively naive method is adopted in both the Vocabulary Levels Test and the Vocabulary Size Test. First, they prepare a word list based on word frequency and then sort words using the word list in the ascending order of word frequency. They then make groups of words, typically consisting of $1,000$ words of similar difficulty. They randomly sample 10 words from each group. As they use a word list consisting of $14,000$ words, there are 140 words sampled to be tested in total.

The Vocabulary Size Test is later thoroughly validated using the Rasch model in (Beglar, 2010). David Beglar is a co-author of the Vocabulary Size Test (Nation and Beglar, 2007). We look deeper into this validation in §4 because we need to introduce the Rasch model before explaining (Beglar, 2010).

3.3 Yes/No Test

This section introduces the "Yes/No Test", a series of works on English vocabulary tests by Paul Meara. The concept of this test first appeared in (Meara and Buxton, 1987) and was then realized as a concrete test format for English language teachers who are not necessarily familiar with vocabulary studies as "Eurocentres vocabulary tests" or "EFL Vocabulary Tests" in (Meara, 1992). Its second edition became available in (Meara, 2010). Although it is called the second edition, Meara (2010) says that it is "essentially the same as the original versions, except that the format has been improved to take account of 2010 typographical conventions."

There are two notable characteristics of the Eurocentres vocabulary test: first, it adopts a **self-**

NOT

PUBLIC

Fig. 3.3. Example of EFL Vocabulary Tests. This page is taken from 51, (Meara, 2010).

report YES/NO format, and second, it extensively uses **pseudowords**. Figure 3.3 shows an example of this test. Unlike a multiple-choice format, where test-takers are instructed to select the option that they think is correct from multiple choices-namely, **four** options for the Vocabulary Size Test - in Figure 3.3, we can see that there is no option for test-takers to select for each item. Instead, they simply have to *report* whether they knows the word or not (instead of *selecting* the correct option). This type of test format is called a self-report format.

Of course, simply replying on the test-takers' self-report opens the door to the possibility of *cheating*: some test-takers may report "yes" to all the items to make their vocabulary size appear bigger than it actually is. The test must be robust against this type of cheating. To make it robust,

NOT

PUBLIC

Fig. 3.4. Example of EFL Vocabulary Tests' testcode. This page is taken from 60, (Meara, 2010).

pseudowords are introduced.

Pseudowords are words that do not exist in the target language. Unlike real words, answering "yes" to pseudowords can reduce the estimated size of the test-takers' vocabulary. If the test-takers answer "yes" to all the items, they end up answering "yes" to pseudowords as well, which reduces their estimated vocabulary size. In "Yes/No" tests, pseudowords are also provided, as shown in Figure 3.4.

Table. 3.2. A stimulus-response matrix. Imported from Figure 1 of Huibregtse et al. (2002) with modifications on notation.

		response	
		YES	**NO**
stimulus	**word**	$p(\textbf{YES}\|\textbf{word})$	$p(\textbf{NO}\|\textbf{word})$
	pseudoword	$p(\textbf{YES}\|\textbf{pseudoword})$	$p(\textbf{NO}\|\textbf{pseudoword})$

3.4 Formulae for Vocabulary Size Estimation using Pseudowords

We have explained that answering "Yes" in the Yes/No test format can reduce the estimated vocabulary size. Then, how much should we reduce the estimated vocabulary size? For this problem, there have been a number of formulae proposed in the literature. These formulae are well surveyed in (Huibregtse et al., 2002; Eyckmans, 2004; Pellicer-Sànchez and Schmitt, 2012). We introduce the most popular ones among the proposed, following (Huibregtse et al., 2002).

First, we introduce notation. Although the argument discussed here is based on Huibregtse et al. (2002), note that the notation is significantly different from that of Huibregtse et al. (2002) in order to correlate with the information in the other chapters. We define $x \in \mathcal{X}$ as the random variable for the stimulus and $y \in \mathcal{Y}$ as the random variable for the stimulus response. In a Yes/No test format, both $\mathcal{X}$ and $\mathcal{Y}$ are binary:

$$\mathcal{X} = \{\textbf{word}, \textbf{pseudoword}\}, \tag{3.1}$$

$$\mathcal{Y} = \{\textbf{YES}, \textbf{NO}\}. \tag{3.2}$$

If the notation is not confusing, we abuse the notation by omitting the random variable; i.e., we simply write $p\left(\textbf{YES}|\textbf{word}\right)$ instead of $p\left(y = \textbf{YES}|x = \textbf{word}\right)$. A confusion matrix, or a *stimulus-response matrix* if being specific to this task, is then shown in Table 3.2.

The probabilities shown in the confusion matrix in Table 3.2 are *true probabilities*. That is, they are to be estimated from the *observed* numbers using a model. We write the observed number of **word** stimuli as $N_{\textbf{word}}$ and the observed number of the **YES** responses for the given **word** stimuli as $N_{\textbf{word},\textbf{YES}}$, and so on. Note that we use the wording *observed* not only for responses but also for stimulus here because we are referring to the *observed* result of a test. We write p_{obs} for observed probabilities. For example,

$$p_{obs}(\mathbf{YES}|\mathbf{word}) \overset{\text{def}}{=} \frac{N_{\mathbf{word},\mathbf{YES}}}{N_{\mathbf{word}}}.$$

Then, we introduce *hit* and *false alarm*.

$$\text{The number of the observed } \textit{hit}: N_{\text{hit}} \overset{\text{def}}{=} N_{\mathbf{NO},\mathbf{word}}, \tag{3.3}$$

$$\text{The number of the observed } \textit{false alarm}: N_{\text{false alarm}} \overset{\text{def}}{=} N_{\mathbf{YES},\mathbf{pseudoword}}, \tag{3.4}$$

$$\text{The number of the observed } \textit{miss}: N_{\text{miss}} \overset{\text{def}}{=} N_{\mathbf{NO},\mathbf{word}}, \tag{3.5}$$

$$\text{The number of the observed } \textit{correct rejection}: N_{\text{correct rejection}} \overset{\text{def}}{=} N_{\mathbf{NO},\mathbf{pseudoword}}. \tag{3.6}$$

3.4.1 Simple Scoring

One of the simplest methods for scoring a Yes/No test format is to reduce the number of hits by the number of false alarms, as shown in (3.7).

$$N_{\text{simple}} \overset{\text{def}}{=} N_{\text{hit}} - N_{\text{false alarm}} \tag{3.7}$$

(3.7) has been criticized as too simple for practical usage by many studies such as (Zimmerman et al., 1977). For example, if there are more false alarms than hits, we can easily see that (3.7) becomes a negative value, making its meaning difficult to interpret.

Another problem is that this is not a probabilistic measure: that is, the result is sensitive to the number of total items or the number of pseudowords, which we may want to change.

3.4.2 The Number of Correct Responses

Another one of the simplest methods for scoring a Yes/No test is to count the number of correct answers, as shown in the following formula:

$$N_{\text{correct response}} \overset{\text{def}}{=} N_{\text{hit}} + N_{\text{correct rejection}}. \tag{3.8}$$

One notable problem in this simple formula is that it handles both the number of hits and the number of correct rejections equally. For example, the test-taker can obtain the score of N_{word} at least by a simple cheating strategy: answering "Yes" to all the items. Even if a serious test-taker managed to get a hit of $N_{word} - 10$ with 10 correct rejections, this formula ends up scoring the latter, serious test-taker the same as the former, cheating test-taker. Moreover, this scoring is not probabilistic, either.

3.4.3 Correction for Guessing

We have seen two naive methods to correct the estimated size of vocabulary. As a more sophisticated method, correction for guessing (frequently abbreviated to "cfg") was introduced into the literature. One of the developers of the Yes/No Test, Paul Meara, used this method in an early publication (Meara and Buxton, 1987). According to Huibregtse et al. (2002), before Meara and Buxton (1987), Anderson and Freebody (1983) also used this method.

This method is derived from the basics of signal detection theory (SDT) (Wickens, 2001; McNicol, 2005). Among the many models that the signal detection theory provides, this method is derived from the high threshold model, although many papers do not mention this model by name. The high threshold model is illustrated in Figure 3.5. Note that the names of random variables and constants are modified so that they fit the vocabulary estimation issues that we are currently focusing on.

The high threshold model introduces another random variable into the process of test-takers responding towards stimuli. In vocabulary testing, this random variable is to denote the **mental state** of a test-taker when he or she is provided with stimuli. We define the random variable **mental state** as $\textbf{mental state} \in \{\textbf{decisive}, \textbf{unsure}\}$.

The decision process in Figure 3.5 can be explained as follows. When the test-taker is provided with a **word** stimuli and forced to answer "YES" or "NO", if he/she is decisive, or *sure* to know the word, he/she answers "YES". If he/she does not know the word, or is *unsure* of the answer, he/she begins a *guessing process*. The probability of being *sure* given a **word** stimulus can be written as $p(\textbf{mental state} = \textbf{sure}|x = \textbf{word})$.

When the test-taker is in the guessing process, he/she answers randomly.

- He/she answers "YES" with the probability of $p(\textbf{YES}|\textbf{unsure})$.
- He/she answers "NO" with the probability of $p(\textbf{NO}|\textbf{unsure})$.

In Figure 3.5, if the given stimulus is a **pseudoword** stimulus, we can see that the test-taker goes into the *same* guessing process. Here is where the fundamental assumptions of this high threshold model lie.

Always unsure of pseudowords The test-taker is assumed to be always **unsure** of pseudowords. In the other words, the test-taker is never sure that the given stimulus is a certain **word**. This is unrealistic in cases in which a pseudoword has a spelling that closely resembles a real word. For example, suppose the stimulus "invinciblity". This is a pseudoword because the “i” after “b” is omitted, although its spelling is similar to the real word “invincibility”. If the test-taker is careless and sure to answer “YES” by mistaking the pseudoword with the real word “invincible”, the test-taker’s decision process is outside the scope of this high threshold modeling.

Single **unsure** state There is one single **unsure** state in the high threshold model. This means that the test-taker enters exactly the same single **unsure** state regardless of whether the given stimulus is a **word** or a **pseudoword**. This is unrealistic in that the *sureness* or the confidence of the test-taker is binary, not gradual. For example, when given a **word** stimulus, the test-taker might think “*I saw the word before, but I don’t remember the meaning. What should I answer?*” This kind of guessing is not handled in the high-threshold model. Huibregtse et al. (2002) calls this complex process of answering “sophisticated guessing”.

In the high threshold model, $p\left(\textbf{sure}|\textbf{word}\right)$ can be used as the test-taker’s score because this value represents the probability that the test-taker is sure given the stimulus is **word**. This value can be estimated exactly as follows in the high threshold model. First, $p\left(\textbf{YES}|\textbf{word}\right)$ can be written as:

$$\begin{aligned}
p\left(\textbf{YES}|\textbf{word}\right) &= p\left(\textbf{YES}|\textbf{sure}\right)p\left(\textbf{sure}|\textbf{word}\right) + p\left(\textbf{YES}|\textbf{unsure}\right)p\left(\textbf{unsure}|\textbf{word}\right) && (3.9)\\
&= p\left(\textbf{sure}|\textbf{word}\right) + p\left(\textbf{YES}|\textbf{unsure}\right)p\left(\textbf{unsure}|\textbf{word}\right) && (3.10)\\
&= p\left(\textbf{sure}|\textbf{word}\right) + p\left(\textbf{YES}|\textbf{unsure}\right)\left(1 - p\left(\textbf{sure}|\textbf{word}\right)\right). && (3.11)
\end{aligned}$$

Here, we use $p\left(\textbf{YES}|\textbf{sure}\right) = 1$, which can be obtained from Figure 3.5, because there is no arrow from **sure** to **NO**. Because $p\left(\textbf{YES}|\textbf{word}\right)$ can be observed and obtained as $p\left(\textbf{YES}|\textbf{word}\right) \approx \frac{N_{\text{false alarm}}}{N_{\textbf{word}}}$, we can obtain $p\left(\textbf{sure}|\textbf{word}\right)$ if we can obtain $p\left(\textbf{YES}|\textbf{unsure}\right)$.

We can obtain $p\left(\textbf{YES}|\textbf{unsure}\right)$ by decomposing $p\left(\textbf{YES}|\textbf{pseudoword}\right)$ as follows:

$$\begin{aligned}
p\left(\textbf{YES}|\textbf{pseudoword}\right) &= p\left(\textbf{YES}|\textbf{sure}\right)p\left(\textbf{sure}|\textbf{pseudoword}\right) + p\left(\textbf{YES}|\textbf{unsure}\right)p\left(\textbf{unsure}|\textbf{pseudoword}\right)\\
&= p\left(\textbf{YES}|\textbf{unsure}\right). && (3.12)
\end{aligned}$$

Thus, we can obtain $p\left(\textbf{YES}|\textbf{unsure}\right)$ as follows:

$$p\left(\textbf{YES}|\textbf{unsure}\right) = p\left(\textbf{YES}|\textbf{pseudoword}\right) \approx \frac{N_{\text{false alarm}}}{N_{\textbf{pseudoword}}}. \quad (3.13)$$

So far, we can see that there are two important observable probabilities in the high threshold model. They are defined as $p_{\text{false alarm}} \overset{\text{def}}{=} p\left(\textbf{YES}|\textbf{pseudoword}\right)$ and $p_{\text{hit}} \overset{\text{def}}{=} p\left(\textbf{YES}|\textbf{word}\right)$. The former is called the *false alarm rate* and the latter is called the *hit rate*. Their probabilities are observable, as follows:

$$p_{\text{hit}} \approx \frac{N_{\text{hit}}}{N_{\textbf{word}}} \tag{3.14}$$

$$p_{\text{false alarm}} \approx \frac{N_{\text{false alarm}}}{N_{\textbf{pseudoword}}} \tag{3.15}$$

Finally, by using (3.11) and (3.13), we can obtain the test-taker's score, $p\left(\textbf{sure}|\textbf{word}\right)$, as follows:

$$p\left(\textbf{sure}|\textbf{word}\right) = \frac{p\left(\textbf{YES}|\textbf{word}\right) - p\left(\textbf{YES}|\textbf{pseudoword}\right)}{1 - p\left(\textbf{YES}|\textbf{pseudoword}\right)} = \frac{p_{\text{hit}} - p_{\text{false alarm}}}{1 - p_{\text{false alarm}}}. \tag{3.16}$$

(3.16), which is called correction for guessing, corresponds to equation "2)" in (Huibregtse et al., 2002). In Pellicer-Sànchez and Schmitt (2012), this value is denoted simply by "cfg".

Although (3.16) seems sophisticated compared to the simple formulae that we saw in the previous subsections, it still can NOT deal with cheating in the form of answering "YES" to all items. This can be easily seen by considering a special case in (3.16). If the test-taker answers "YES" to all the items, $p_{\text{hit}} \approx 1$ in (3.16) [*1]. If $p_{\text{hit}} \approx 1$, as the numerator and the denominator are almost equal in (3.16), $p\left(\textbf{sure}|\textbf{word}\right)$ is estimated to be almost 1, the perfect score, regardless of the false alarm rate $p_{\text{false alarm}}$. Thus, as (3.16) is not robust against test-taker cheating, we will introduce a more robust measure in the following subsections.

We can also see that the value of (3.16) can NOT to be computed when both $p_{\text{hit}} = 1$ and $p_{\text{false alarm}} = 1$ exactly. However, this characteristic does not seem to be much of a problem. Both Meara's Δ and I_{sdt}, the two correction formulae that we will introduce in this section, have this characteristic. The reason is presumably that this kind of perfect score rarely happens and can be detected very easily.

3.4.4 Meara's Δ

In the previous subsection, we derived the correction for guessing, or the cfg method, but we also saw its shortcomings, namely, that the method is not robust against simple cheating in the form of answering "YES" to all items. To this end, Meara (1992) used the following formula, a formula with a discounting term added to the cfg method's formula (3.16), as follows:

$$p_{\text{meara}} = \frac{p_{\text{hit}} - p_{\text{false alarm}}}{1 - p_{\text{false alarm}}} - \frac{p_{\text{hit}}}{p_{\text{false alarm}}} = p_{\text{cfg}} - \frac{p_{\text{hit}}}{p_{\text{false alarm}}} \tag{3.17}$$

By comparing (3.16) and (3.17), we can see that the two are essentially the same except that the discounting term $-\frac{p_{\text{hit}}}{p_{\text{false alarm}}}$ is added in the latter. We can easily see that this discounting term

[*1] Here, we use $\approx$ in the sense that $p_{\text{hit}} < 1$ but is very close to 1.

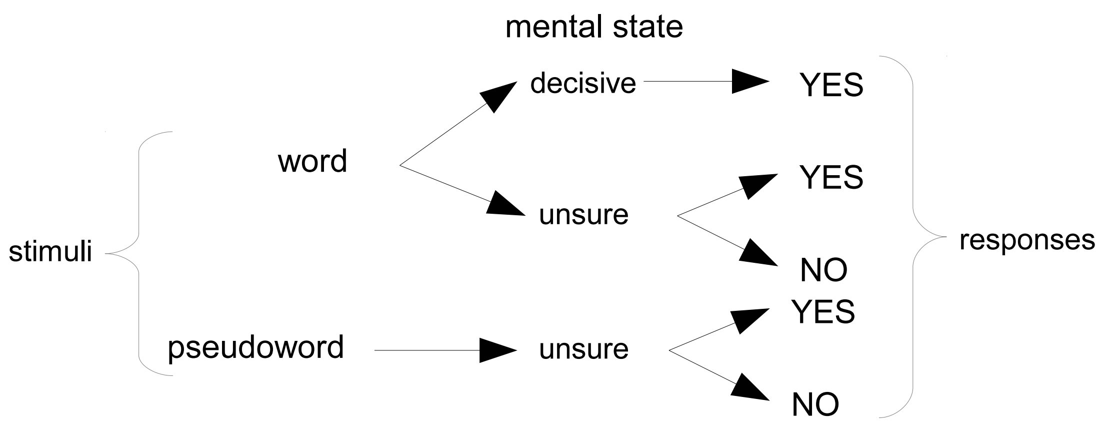


Fig. 3.5. The high threshold model.

is 0 when the test-taker answers perfectly, i.e., “YES” to all the **word** stimuli and “NO” to all the **pseudoword** stimuli; in this case, because $p_{\text{false alarm}} = 0$ and $p_{\text{hit}} = 1$, the discounting term $-\frac{p_{\text{hit}}}{p_{\text{false alarm}}} = 0$, and thus nothing is discounted, and $p_{\text{meara}} = p_{\text{cfg}} = 1$.

In contrast, if the test-taker cheats by answering “YES” to all the items, while $p_{\text{cfg}} = 1$ as we saw in the previous subsection, the discounting term becomes 1 as both the numerator $p_{\text{false alarm}} = 0$ and the numerator $p_{\text{hit}} = 1$ in the discounting term of (3.17). This means that the discounting term penalizes (3.16) so greatly that p_{meara} ends up as 0. In this way, (3.17) is robust against simple cheating.

(3.17) is called Meara’s Δ because this calculation method was published with the first version of the Eurocentres vocabulary test, a Yes/No format test that Paul Meara himself devised in 1992 (Meara, 1992). As the simple cheating strategy of answering “YES” to all the items is easy to come up with when test-takers are instructed to take a self-report format test, it can be said that the invention of Meara’s Δ greatly helped the widespread use of Yes/No format tests, namely the Eurocentres vocabulary test.

The derivation of (3.17) is not heuristic, whereas the discounting term added to (3.16) seems to be so. Its derivation is based on the calculation of the area under the curve (AUC) of the receiver operating characteristic (ROC) curve, although we do not go into the derivation deeply because it is beyond our focus. The derivation of (3.17) is thoroughly detailed in (Huibregtse et al., 2002).

3.4.5 I_{sdt}

In the previous subsection, we saw that Meara’s Δ is robust against the simple cheating strategy of test-takers answering “YES” to all items. However, Huibregtse et al. (2002) points out that

Meara's Δ has another problem: when the p_{hit} is fixed and the $p_{\text{false alarm}}$ is rising, its score decays too quickly, and goes into negative values.

$$I_{sdt} = 1 - \frac{4p_{\text{hit}}\left(1 - p_{\text{false alarm}}\right) - 2\left(p_{\text{hit}} - p_{\text{false alarm}}\right)\left(1 + p_{\text{hit}} - p_{\text{false alarm}}\right)}{4p_{\text{hit}}\left(1 - p_{\text{false alarm}}\right) - \left(p_{\text{hit}} - p_{\text{false alarm}}\right)\left(1 + p_{\text{hit}} - p_{\text{false alarm}}\right)} \tag{3.18}$$

As with Meara's Δ, I_{sdt} is not derived from heuristics.

3.4.6 Negative Aspects on Pseudowords

We have surveyed studies on Yes/No tests, which use pseudowords extensively. However, many studies argue that using pseudowords is unnecessary. In this subsection, we introduce some negative aspects of the use of pseudowords in vocabulary tests.

First, it is difficult to create good pseudowords in the first place. A pseudoword should never have been used as a word in any dialect of a language. To take English as an example, a pseudoword should be spelled in a way that is realistic for English words. This makes it difficult for non-native speakers to create a language test.

Moreover, the responses to pseudowords are greatly affected by the test-takers' native language, as has been admitted by Meara (1992) himself. For example, the vocabulary size of speakers of romance languages (such as French, Italian, Spanish, and Greek) was initially overestimated because the test-takers knew the words through their native languages as these languages share many cognate words with English low-frequency words. For example, take the word "emancipate". To reduce the overestimation, Meara (1992) states that they deliberately created pseudowords from Greek and Latin roots so that test-takers of these languages would tend to answer "YES" to the pseudowords. While they report that this creation of pseudowords did not affect the results of test-takers of other native languages, we can interpret this to mean that a detailed and accurate understanding of both the target language and the test-takers' native language is required to create pseudowords. Indeed, Meara (1992) also states that "French speakers seem to be more willing to accept imaginary words than speakers of other languages" and they could not find any reason for this at the time.

Second, many studies report that pseudowords are unnecessary or sometimes even harmful in estimating vocabulary size. Studies regarding this issue are well surveyed in (Stubbe, 2012). Mochida and Harrington (2006) administered a Vocabulary Levels Test (multiple-choice format) and EFL Vocabulary Size Test (yes/no format) to the same test-takers and compared the vocabulary size of both tests. They found that simple "hits", i.e., the number of real words to which the test-takers answered "YES", in the EFL Vocabulary Size Test could best predict the vocabulary

Table. 3.3. Dale's scale.

Stage 1	"I never saw it before."
Stage 2	"I've heard of it, but I don't know what it means."
Stage 3	"I recognize it in context - it has something to do with ..."
Stage 4	"I know it."

Table. 3.4. Paribakht & Wesche's elicitation scale.

I	"I don't remember having seen this word before."
II	"I have seen this word before, but I don't know what it means."
III	"I have seen this word before, and I think it means (synonym or translation)."
IV	"I know this word. It means (synonym or translation)."
V	"I can use this word in a sentence: (Write a sentence.) (If you do this section, please also do Section IV)

size measured by the Vocabulary Size Test. Mochida and Harrington (2006) concluded that "the presence of nonwords had little effect on their test performance".

Thus, it would be wise to approach the use of pseudowords with caution. We summarize these surveys as follows:

- If test makers are not native-speakers of the target second language, they may have to rely on a list of pseudowords created by native speaker researchers.
- Completely omitting pseudowords is preferable to using low-quality pseudowords.

3.5 Vocabulary Knowledge Scale

In Read (2000), two methods are introduced to make questions for measuring the degree of human language ability.

Multiple-choice items A question consists of one word and multiple (usually four) answer choices. The subjects are told to choose the choice that has the same meaning as the word.

Self-report scale A subject answers with freely chosen words indicating how well he/she knows the word in his/her subjective judgment.

Table. 3.5. Word Samlping Sources.

Name of Test	Sampling Sources
Vocabulary Size Test Nation (2012b)	"The words to be tested are sampled from the British National Corpus word family lists which now go up to the 25th 1000. The British National Corpus word lists (more recently the BNC/COCA wordlists) are lists of word families developed for two main purposes."
EFL Vocabulary Test Meara (1992)	"Two different word lists were used as the source for these tests. The Level 1 and Level 2 tests are based on Paul Nation's Vocabulary lists: words, affixes and stems (Nation 1986). Levels 3, 4 and 5 are based on Hindmarsh's Cambridge English Lexicon (Hindmarsh 1980). The final level A tests are also based on work by Nation."

We can see that the three tests to measure written receptive vocabulary size, namely the Vocabulary Levels Test, the Vocabulary Size Test, and the Yes/No Test, are based on individual word frequency lists. The words to be tested are sampled using a single word frequency list. It is notable that no test tries to handle multiple word frequency lists at a time. Instead, each test makes a substantial effort to create a solid word frequency list on which the test is to be based. The word frequency lists used in each corpus are summarized in Table 3.5.

3.6 Word Frequency List

We can see that the three tests to measure written receptive vocabulary size, namely the Vocabulary Levels Test, the Vocabulary Size Test, and the Yes/No Test, are based on each word frequency list. The words to be tested are sampled using a single word frequency list. It is notable that no test tries to handle multiple word frequency lists at a time.

Instead of handling multiple word frequency lists, each test makes substantial effort to create a good word frequency list on which the test is to be based. Word frequency lists used in each corpus are summarized Table 3.5.

We can see that the British National Corpus (BNC) The BNC Consortium (2007) and the Corpus of Contemporary American English (COCA) Davies (2011) are basically used as word sampling sources in the Vocabulary Size Test. However, the word list is not sampled from these corpora as is. In Nation (2012a), Nation stated that "Six million tokens of this corpus were spoken English from both British and American English (see Corpus/PN corpus for 2000) as well as movies and TV programs". The first 2,000 words are sampled from this specially made corpus and the latter are sampled from BNC and COCA after removing the first specially made $2,000$-word list.

Likewise, Meara (1992) takes special care for the first, or easy, part of the word list. "Nation 1986" is another specially designed word list by Paul Nation, and we can see that Paul Meara used it. Meara (1992) states that "(Hindmarsh 1980)" is basically revision of West (1953) and he was unwilling to use the list but he ended up with using it because he could not find an alternative.

We can say that, although the word lists used in vocabulary testing are sampled from general corpora such as BNC and COCA, studies carefully make the easy parts of the word list so that they include words necessary for daily communication.

3.7 Other studies

Up to this point, we have surveyed vocabulary studies on second language. However, there are other studies regarding human vocabulary in general. In this section, we briefly introduce some other fields related to second language vocabulary.

3.7.1 Vocabulary and Reading

While we discussed many vocabulary test studies, determining the purpose of vocabulary testing concretely in the beginning is essential for designing vocabulary tests, as we mentioned in §3.1. Otherwise, we cannot be sure what we are measuring.

Paul Nation's group, the team that developed the Vocabulary Size Test, seem to be quite sensitive on the purpose of their test. They clearly state that it "is designed to measure both first language and second language users' written receptive vocabulary size in English" in (Nation, 2012b). They also studied and surveyed the relation between vocabulary size and reading.

The method of studying the relation between vocabulary size and reading is basically simple: the test-takers are subjected to both the Vocabulary Size Test and reading comprehension tests, and then their correlation is determined.

The relationship between vocabulary knowledge and text readability has been thoroughly studied by educational experts (Nation, 2006).

3.7.2 Mental Lexicon Studies

The vocabulary test studies that we have surveyed in this chapter (such as the Vocabulary Size Test) belong to the field of Teaching English as a Second Language (TESOL), which is generally more practical than the field of Second Language Acquisition (SLA). SLA studies are more theoretical in terms of the human cognitive viewpoint. Both TESOL and SLA are sub-fields of applied linguistics.

In the field of SLA, there is another branch of vocabulary studies called the mental lexicon. Mental lexicon studies, which are closely related to *cognitive science*, explore the structure of human memory of vocabulary and how humans access their vocabulary. Because of this, for example, mental lexicon studies mainly focus on *response time* rather than vocabulary size. Mental lexicon itself is not a concept limited to second languages: there are mental lexicon studies on native languages too, as we will see in the next subsection.

3.7.3 Vocabulary Studies on Human Native Language

Up to now we have introduced vocabulary studies on second languages, but there are also studies on the vocabulary of the first (i.e., native) language. Note that, because the target of these native language studies is different from those of second languages, the purpose of the studies is quite different. Therefore, the comparison of these studies with the vocabulary studies that we have seen in the previous subsections is not straight-forward. Here, we introduce a few studies by categorizing their purposes or the motivation behind them.

One of the purposes of vocabulary in native language studies is to measure how large the vocabulary size of a human is. For example, "Does the vocabulary size of the native language largely differ by language?" or "How much does education affect the vocabulary size of the native language?" are typical research questions in these studies. Another characteristic is that they are keen on how the stimuli are given to the humans: by auditory means, by visual means, or both. Unlike a second language, a native language is acquired auditorily first, and then visually. In other words, in native languages, the acquisition of listening and speaking comes earlier than reading and writing. Therefore, auditory stimuli are studied as well as visual stimuli. For example, Amano and Kondo (1998) call this auditory-or-visual distinction *modality* and studied the difference of vocabulary size in the Japanese mental lexicon of Japanese native speakers by stimulus modalities. They found that there is little difference in the size by modalities.

The vocabulary size of children, or *developmental vocabulary*, has also been extensively studied. In computational linguistics, Kireyev and Landauer (2011) proposed an extension of word difficulty called “word maturity” by focusing on the analysis of child development in terms of native language. Their extension aimed to “track the degree of knowledge of each word at different stages of language learning” using latent semantic analysis. Thus, both their purpose and the method of extending word difficulty differ from ours.

3.8 Chapter Summary

In this chapter, we surveyed a variety of vocabulary assessment studies. Vocabulary assessment can be categorized in a response modeling from stimuli in Table 1.1. The majority of the effort in vocabulary assessment has been to measure second vocabulary size. However, we can say that the *vocabulary size* in these studies is used as a convenient measure of language proficiency rather than an actual measure of vocabulary size. Indeed, as far as we surveyed, we could not find a study that actually and exhaustively checks whether a user knows a word or not for a large number of words, while we can find such exhaustive studies in the mental lexicon study of native languages.

It is also noteworthy that most studies stressed the importance of the purpose of a test, since it defines what a test measures: a test cannot measure what it does not aim to measure. This means that the vocabulary knowledge that vocabulary tests can measure and that support systems can measure by using our framework may be different: in other words, in support systems, users may act differently than they do in language tests.

Chapter 4

Item Response Theory and Logistic Regression

This chapter introduces the item response theory (IRT) and its relation to the logistic regression. To let the readers grasp the overview of the models that we introduce in this thesis, we illustrate an overview of the containment relationships of the models in Figure 4.1.

In Figure 4.1, the hashed area corresponds to the **Rasch model**, the most basic model. All the other models contain this model as a special case. For example, the Rasch model is a special case of the IRT, which is a collective term of three models: the three-parameter, two-parameter, and one-parameter models. The two-parameter model is a special case of the three-parameter model and the one-parameter model is a special case of the two parameter model. The Rasch model is an alias of the one-parameter model.

Another extension of the Rasch model is the shared difficulty model. Note that the shared difficulty model is not a special case of IRT. The proposed convex model we introduce in §5 corresponds to an extension of the shared difficulty model. Both the Rasch and shared difficulty models can be expressed as a special case of logistic regression.

We explained what desirable property each model has in Table 1.2.

We begin this chapter by explaining the three-parameter IRT model and its relation to the Rasch model.

4.1 Problem setting

Let U be a set of users, and V be a set of vocabulary. We denote the number of users as $|U|$ and the number of words as $|V|$. A datum can be expressed using the triplet (y, u, v). Here,

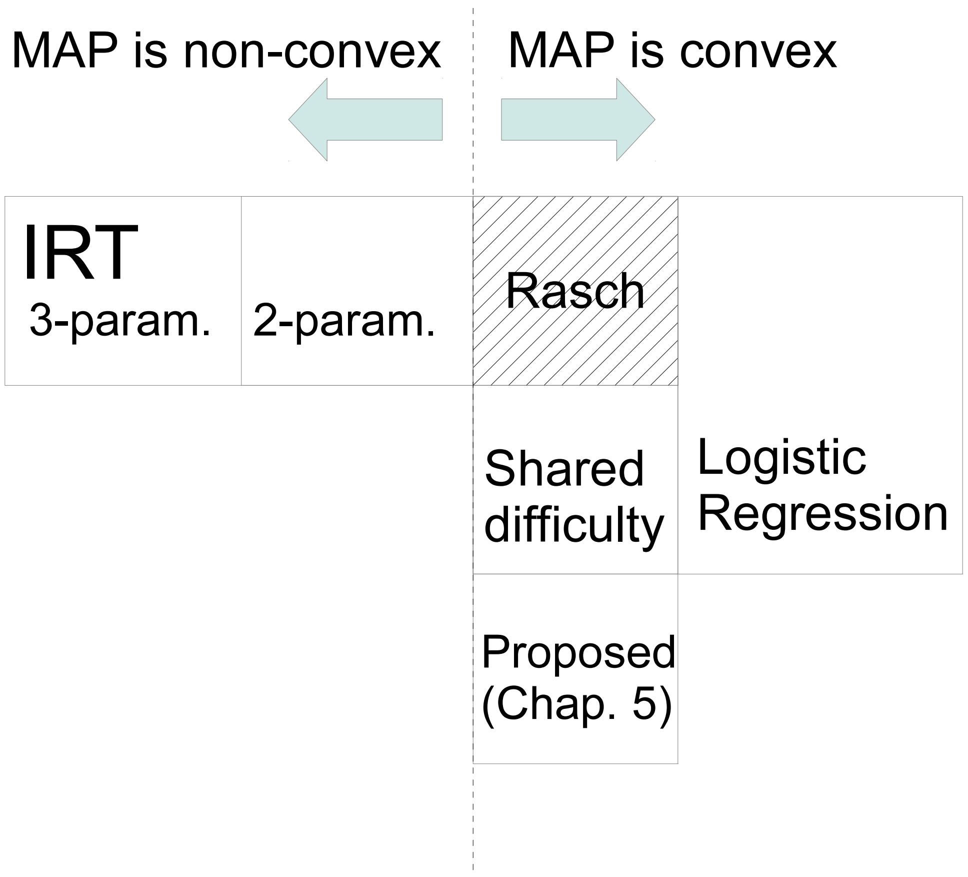


Fig. 4.1. Overview of containment relationships of models. Hashed area is Rasch model. All models except Rasch model contain this model as special case.

$y \in \{-1, 1\}$ is the label denoting whether or not user u knows word v, $(1, u, v)$ means that user u knows word v, and $(0, u, v)$ means he/she does not know word v. Using these notations, the prediction task is defined to predict the label y given (u, v). We denote a dataset of N data as $\mathcal{D} = \{(y_1, u_1, v_1), \ldots, (y_N, u_N, v_N)\}$. Thus, if necessary, we also write $\boldsymbol{x} = (u, v)$. Here, u is the index of user $u \in \{1, \ldots, |U|\}$ and v is the index of word $v \in \{1, \ldots, |V|\}$. The u and v can be interpreted as the unlabeled part of the data as well.

For simplicity, we assume that for one user $u \in U$ and word $v \in V$ pair, there exists only one label y; thus, one datum (y, u, v) in the dataset. This restriction enables us to depict the data set in a matrix form, as shown in Figure 4.2. The rows of the matrix correspond to users and the columns of the matrix correspond to words. For one row (user) and one column (word), there is only one cell; thus, only one label y. With this restriction, N is the number of cells in the matrix.

The dataset we used in the evaluation agrees with this restriction; however, we cannot always assume this restriction in a realistic setting. This is the reason we did not directly jump to matrix-based prediction methods such as low-rank approximation using singular value decomposition. For example, in a realistic dataset, such as word-click logs in a reading support system, contradiction and repetition are common. For contradiction, if both $(1, u, v)$ and $(0, u, v)$ appear in the dataset, it may mean these two datasets are unreliable. Repetition of multiple $(1, u, v)$ may mean that user u is more familiar with word v than just one $(1, u, v)$. Both IRT models and the proposed convex model that we introduce in §5 can naturally handle these cases.

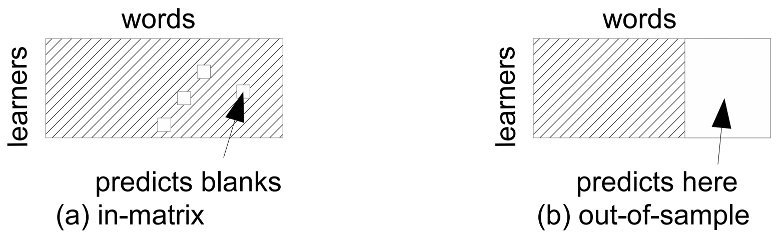


Fig. 4.2. Two problem settings; (a) in-matrix, (b) out-of-sample

Figure 4.2 explains two problem settings: *in-matrix* and *out-of-sample*. The hashed areas in the figure denote the training data. The blank areas denote the test data. In the *in-matrix* setting, the test data are randomly placed in the matrix.

4.2 Item Response Theory

IRT is a statistical method for analyzing the results of tests of human abilities including language tests. The IRT is thought to be used in TOEFL (Test of English as a Foreign Language) and TOEIC (Test of English for International Communication). We will explain the item response theory based on Baker and Kim (2004); Toyoda (2005)[*1].

First, the IRT calculates "parameters" from human ability test results. Once the parameters have been calculated, the IRT can then be used to estimate the following probabilities.

- probability that a user will answer a "different" question correctly
- probability that a "different" user will answer the original question correctly

Here, "different" means that no record with the same pair of user and question is found in the previous test results. This covers the user or question being new and the user or question existing

[*1] Although written in Japanese, we cited Toyoda (2005) and Toyoda (2012) because his series of books are seemingly quite famous in Japanese research communities using IRT. For an English reference, see Baker and Kim (2004) since it also covers the IRT models discussed in this thesis.

but the user not answering that question in the previous test.

4.2.1 Definition of IRT

The IRT involves three concepts: “users”, “items”, and “responses”. Users are simply the set of users, items correspond to the questions in a test, and a response indicates how correctly the user responded to an item.

(4.1) shows the item response theory. (4.1) codes y as a binary random variable (though in some variations of IRT, y can take more than two states); that is, $y = 1$ when user u answers item v correctly and $y = 0$ otherwise.

$$P(y = 1|u, v) = c_v + (1 - c_v)\sigma\left(C_D a_v\left(\theta_u - b_v\right)\right). \tag{4.1}$$

Here, σ is the logistic sigmoid function, which is widely used in probabilistic models because, for a value $x \in \mathcal{R}$, $0 < \sigma(x) < 1$ holds.

$$\sigma(x) = \frac{1}{(1 + \exp(-x))}. \tag{4.2}$$

(4.1), C_D is a constant that is used to make the form of the logistic sigmoid function to that of the cumulative distribution function of the normal distribution whose mean and variance are 0 and 1, respectively.It is known that the difference between the two functions can be minimized when $C_D = 1.7$ Toyoda (2005), thus, this value is used if necessary. However, because we can transform the values of the parameters even after the estimation of the parameters by using this constant factor C_D, many studies such as Baker and Kim (2004) simply set $C_D = 1$ and consider the following formula for simplicity.

$$P(y = 1|u, v) = c_v + (1 - c_v)\sigma\left(a_v\left(\theta_u - b_v\right)\right). \tag{4.3}$$

(4.3) has four parameters to estimate, as listed and explained below.

θ_u	Ability parameter of the user u.
b_v	Difficulty parameter of the item v.
a_v	Discrimination parameter of the item v. This parameter should be positive.
c_v	Guessing parameter of the item v. This parameter should be in the range of $[0, 1]$.

The ability of every user $u \in U$ is modeled by user parameter θ_u. The higher θ_u is, the greater is the ability.

The difficulty of each item $v \in V$ is modeled by b_v. The higher b_v is, the more difficult item v is and the fewer users can answer correctly.

The latter parameters are optional. Discrimination parameter a_v determines how steep the *item characteristic curve* of item v is. The higher a_v is, the steeper the item characteristic curve becomes. We define item characteristic curve in the next subsection.

Guessing parameter c_v determines the probability of test takers guessing the correct answer. This parameter is typically used to model multiple choice items. For example, if a test taker is supposed to choose one item from five choices, then $c_v = \frac{1.0}{5.0} = 0.20$.

4.2.2 Item Characteristic Curves

This section introduces item characteristic curves. Simply speaking, an item characteristic curve is a two-dimensional curve that shows the probability of (4.3) against the user ability parameter θ_u. In the IRT models described in (4.3), there are three *item parameters* for each item: difficulty, discrimination, and guessing, while there is only one parameter for each user: the user ability parameter θ_u. An item characteristic curve shows the relation between item parameters and the user ability parameter. The value of an item characteristic curve ranges from 0 to 1, showing the probability that a user correctly answers an item, i.e., in our application of vocabulary knowledge modeling, the probability that a user knows a word.

Figure 4.3 shows how an item difficulty parameter affects item characteristic curves. We can see that the curve shifts to the right when the item difficulty parameter is $b = 1$ compared to the curve of $b = 0$. This shift to the right means that the item (word) becomes more difficult for a user. For example, let us suppose the user's ability parameter is 0. When $b = 0$, the probability that the user answers correctly is 0.5. If the curve shifts to the right, the probability that the user answers correctly decreases: about 0.3. On the other hand, when the curve shifts to the left at $b = -1$, the probability that the user answers correctly increases: about 0.8. Thus, we can say that the larger the value of difficulty parameter b is, the higher the probability of the user answering the item is.

Figure 4.4 shows how an item discrimination parameter affects item characteristic curves. We can see that the item characteristic curve becomes steeper by the increase in a. The steeper the curve becomes, the more clearly the difference between a high-ability user and a low-ability user becomes. For example, suppose a high-ability user with ability parameter 5. When $a = 1$ and $a = 2$, this user can answer almost perfectly for the item with difficulty 0. However, even for an item with the same difficulty, when $a = 0.5$, the high-ability user fails to answer correctly with some probability. In other words, we cannot simply say "this item is easy for a user with ability

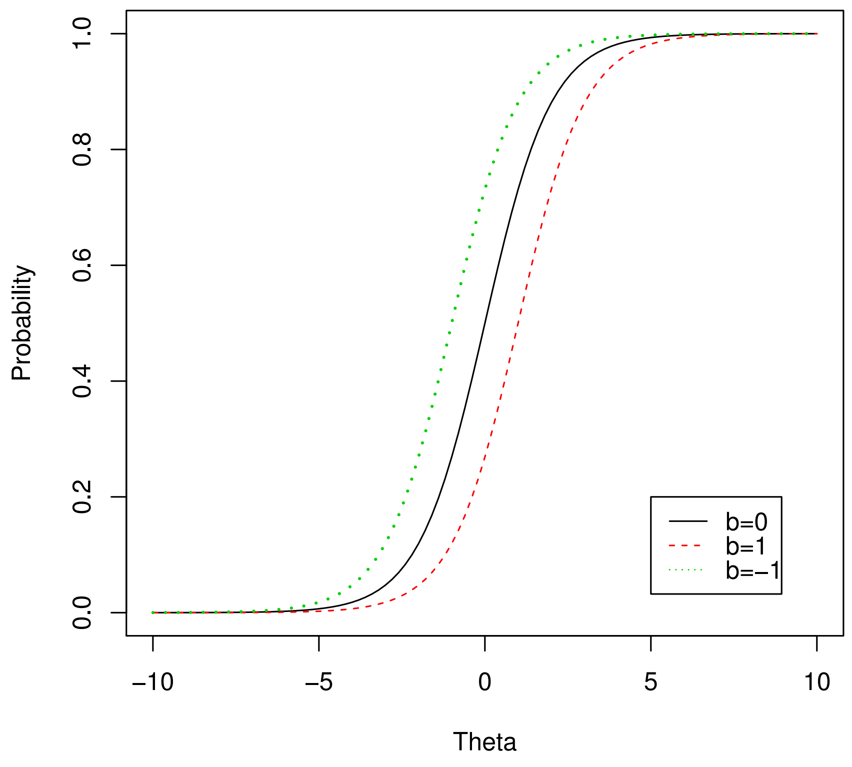


Fig. 4.3. Item characteristic curve by changing value of difficulty parameter b. For other two parameters, $a = 1$ and $c = 0$.

5" when $a = 0.5$.

For another example, if a is close to 0, the probability of answering correctly becomes almost 0.5. In this case, the item (problem) is meaningless in the sense that it does not correlate with an ability parameter. Thus, items (words) with low discrimination parameter values can be regarded as *noisy words*, as introduced in §1.

Finally, we explain the guessing parameter. Figure 4.5 shows how a guessing parameter affects item characteristic curves. The guessing parameter shows the probability of a user to answer correctly by chance. For example, in a multiple-choice question, where users are to select the correct option from 4 confusing options, the user can answer correctly by chance with the probability of 0.25. In this case, we can set $c = 0.25$. Although this parameter can be estimated, it is frequently manually set from the nature of items.

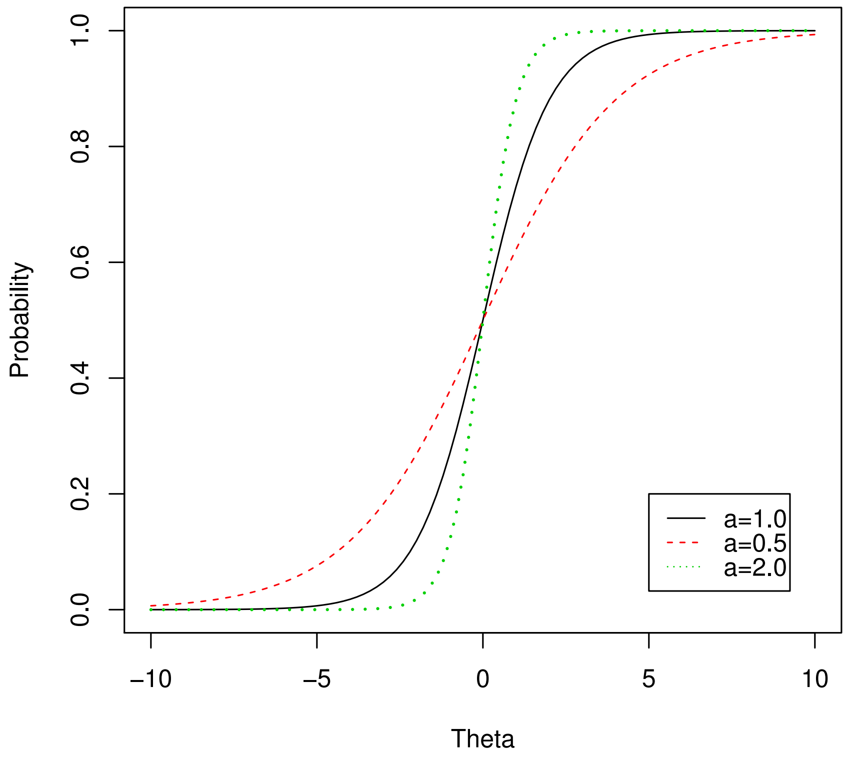


Fig. 4.4. Item characteristic curve by changing value of discrimination parameter a. For other two parameters, $b =$ and $c = 0$.

4.2.3 Rasch Model

$$P(y = 1|u, v) = \sigma\left(\theta_u - b_v\right). \tag{4.4}$$

By comparing (4.3) with (4.4), you can find that (4.4) is a special case of (4.3). Indeed, (4.4) is derived from (4.3) by substituting $a_v = 1$ and $c_v = 0$ for $\forall v \in V$, where $a_v = 1$ means that all the questions are regarded as equally discriminative and $c_v = 0$ means that it is assumed impossible to find the correct answer by guessing.

Figure 4.6 explains how the Rasch model works. This model can be interpreted as predicting a user's vocabulary with the following steps:

1. We rank words according to a measure of word difficulty.

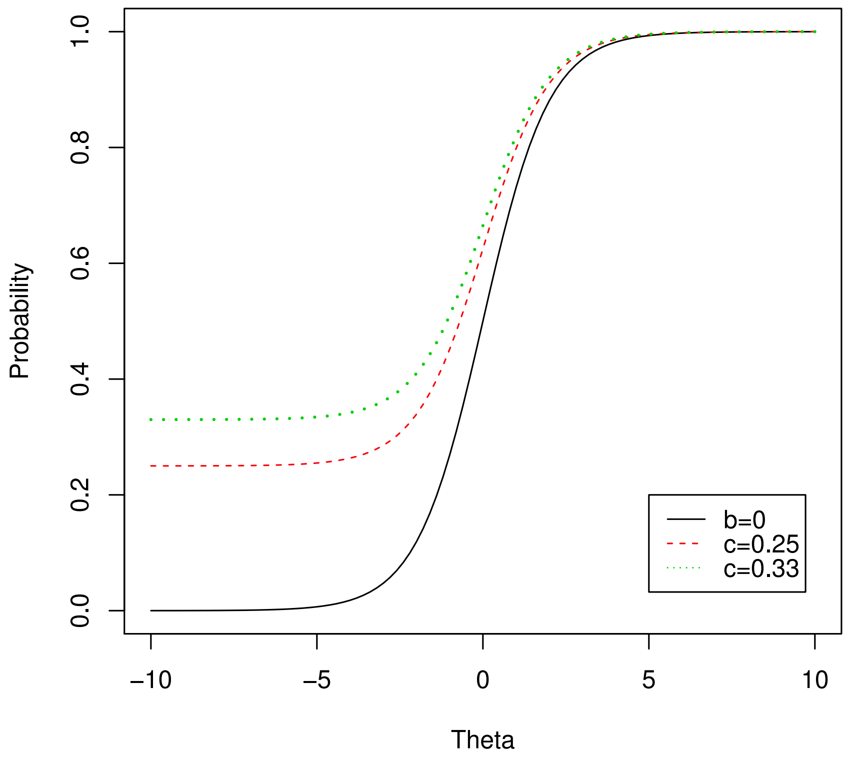


Fig. 4.5. Item characteristic curve by changing value of guessing parameter c. For other two parameters, $a = 1$ and $b = 0$.

2. We decide the threshold for a user.
3. Words with greater difficulty than the threshold are predicted to be unfamiliar to the user, and vice versa.

Although these step seem too simple, it is the core idea of the Rasch model, which has been widely used in language testing.

There are only two kinds of parameters to be trained in the Rasch model:

b_v	the difficulty of word v.
θ_u	the ability of user u.

In the Rasch model, the subtraction of two parameters $\theta_u - b_v$ in (4.4) denotes exactly the same mechanism as the simple vocabulary prediction in Figure 4.6. Here, b_v maps each word v into a point on the axis, and θ_u works as a threshold. When $P(y = 1|u, v) \geq 0.5$, we can assume user

u knows word v. Due to the logistic sigmoid function, when $P(y = 1|u, v) \geq 0.5$ holds true, $\theta_u - b_v \geq 0$, that is, $\theta_u \geq b_v$. Therefore, the Rasch model determines that user u knows all words whose word difficulty b_v is lower than the users' ability θ_u.

The priors for the parameters are usually set as follows:

$$P(\theta_u|\eta_\theta) = \mathcal{N}\left(0, \eta_\theta^{-1}\right) \qquad (\forall u \in U) \tag{4.5}$$

$$P(b_v|\eta_b) = \mathcal{N}\left(0, \eta_b^{-1}\right) \qquad (\forall v \in V), \tag{4.6}$$

where $\mathcal{N}$ denotes the probability distribution function of the normal distribution. Frequently, the hyper parameters η_θ and η_b are set as $\eta_\theta = \eta_b$. If $\eta_\theta = \eta_b$, the parameters, b_v and θ_u of the Rasch model can be obtained using a standard log-linear model solver.

One of the notable problems with the Rasch model is that it does not take into account the *out-of-sample* setting. That is, it cannot predict words that do not appear in the training set. For example, if there is a new word in a document in a reading support system, we need to re-create the training set with the new word for the system to be able to predict new words as well. This restriction makes the application systems using the prediction task quite impractical.

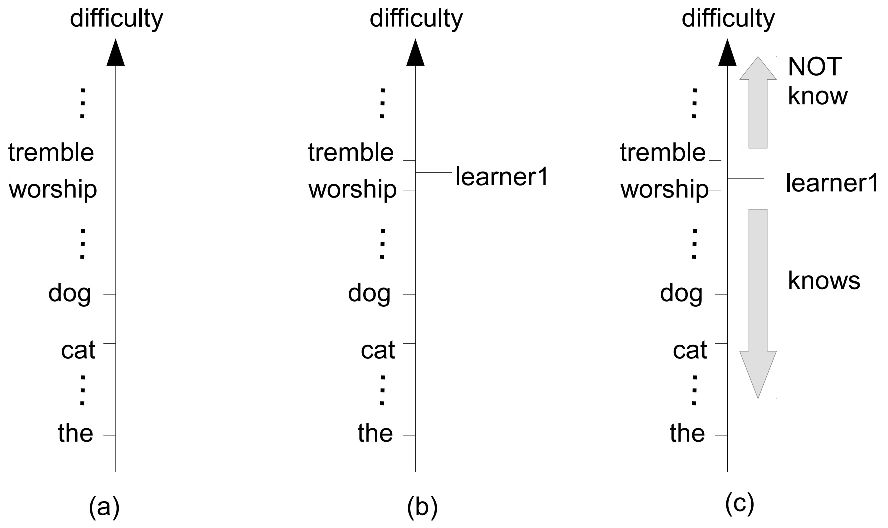


Fig. 4.6. Simple vocabulary prediction model. (a) First, assume there is a difficulty measure that maps each word to a point on the axis of the measure. (b) Second, each user's ability is also mapped to a point on the same axis. (c) Third, the words with the greatest difficulty to the point designating the user's ability is predicted to be unfamiliar to the user, and vice versa.

4.3 IRT and Logistic Regression

This section explains the relation between the IRT and logistic regression. In (4.4), we can see that we can replace the difficulty parameter b_v with $\mathbf{w}^\top \boldsymbol{\phi}(v)$. Here, $\boldsymbol{\phi}(v)$ is a feature vector regarding word v. We write the number of dimensions of $\boldsymbol{\phi}$ as K.

One of the key benefits of this replacement is that, even if there is a new word in the test data and there are words in the training data that share the same features with the new word, the word difficulty of the new word can be obtained by calculating $\mathbf{w}^\top \boldsymbol{\phi}(v)$. The full form of the likelihood becomes the following.

$$P\left(y=1|u,v;\mathbf{w}\right) = \sigma\left(\theta_u - \mathbf{w}^\top \boldsymbol{\phi}\left(v\right)\right). \tag{4.7}$$

Priors for the likelihood (4.7) are set as follows. We call this model the shared difficulty model. This model is used in Ehara et al. (2010).

$$P\left(\theta_u|\eta_\theta\right) = \mathcal{N}\left(0, \eta_\theta^{-1}\right) \qquad (\forall u \in U) \tag{4.8}$$

$$P\left(\mathbf{w}|\eta_w\right) = \mathcal{N}\left(\mathbf{0}, \eta_w^{-1} I\right), \tag{4.9}$$

where I denotes the $K \times K$-sized identity matrix. If we set $\eta_w = \eta_\theta$, this model reduces to a simple l2-norm-regularized logistic regression as Ehara et al. (2010) used. However, they do not mention the out-of-sample setting or the general likelihood.

The idea of replacing b_v with $\mathbf{w}^\top \boldsymbol{\phi}(v)$ was proposed by Fischer (1983), and they call this model the *logistic linear test model*, or **LLTM**. We can say that the shared difficulty model is basically equivalent to LLTM. However, because the details of the two models are different, we do not regard the two models as completely the same. Most importantly, the purpose of replacing b_v with $\mathbf{w}^\top \boldsymbol{\phi}(v)$ is different: LLTM is aimed at better analysis while Ehara et al. (2010) is aimed at better prediction accuracy by using the feature vector $\boldsymbol{\phi}$. For example, **LLTM** limits the features used in $\boldsymbol{\phi}$ to binary features, i.e., taking their values from $0, 1$.

We can also easily see that (4.7) is a special case of the logistic regression. To see this, we concatenate all the users' ability parameters θ_u and write $\boldsymbol{\theta} = \left(\theta_1, \ldots, \theta_{|U|}\right)^\top$. We also introduce a $|U|$ dimensional unit vector $\boldsymbol{e}_u$, whose u-th element is 1 and all the other elements are 0. Then, the linear part inside σ in (4.7), $\theta_u - \mathbf{w}^\top \boldsymbol{\phi}\left(v\right)$, can be expressed as the inner product of a parameter vector $\begin{pmatrix} \boldsymbol{\theta} \\ \mathbf{w} \end{pmatrix}$ and a feature vector $\begin{pmatrix} \mathbf{e}_u \\ -\boldsymbol{\phi}(v) \end{pmatrix}$ as:

$$\begin{pmatrix} \boldsymbol{\theta} \\ \mathbf{w} \end{pmatrix}^\top \begin{pmatrix} \boldsymbol{e}_u \\ -\boldsymbol{\phi}(v) \end{pmatrix} = \boldsymbol{\theta}^\top \mathbf{e}_u - \mathbf{w}^\top \boldsymbol{\phi}(v) = \theta_u - \mathbf{w}^\top \boldsymbol{\phi}(v) .$$

This expression is, by definition, the binary logistic regression, whose probability is expressed as the inner product of its parameter vector and its feature vector.

Expression using the concatenated forms such as $\begin{pmatrix} \boldsymbol{\theta} \\ \mathbf{w} \end{pmatrix}$ and $\begin{pmatrix} \mathbf{e}_u \\ -\phi(v) \end{pmatrix}$ is sometimes cumbersome. We may want to express the parameter vector and the feature vector using a single variable, especially to express the update formulae for parameter estimation. Thus, in this thesis, if what is meant is clear, we may abuse $\mathbf{w}$ and $\boldsymbol{\phi}$ to express the concatenated forms.

4.4 Parameter Estimation of IRT

In the beginning of this chapter, we mentioned that the maximum-a-posteriori two-parameter IRT and the three-parameter IRT are non-convex. In this section, we explain how the parameters of these models are estimated.

Unlike other articles and books introducing parameter estimation methods of the IRT, we want to introduce them from the viewpoint of *convexity*. The process of parameter estimation can be expressed in a form of optimization problems. If the optimization problems are convex, then, theoretically, we can obtain the global optimum parameters, which are good for practical application systems on which this thesis is focused.

4.4.1 Independence Assumption

Following Baker and Kim (2004), we point out that there underlies three assumptions on the independence of variables in (4.10).

Independence of Users Users are assumed to be randomly sampled from the population; therefore, they are assumed to be mutually independent. For example, this assumption omits the case in which the users cooperate to solve problems (i.e., items).

Independence between Item Parameters and User Parameters Item parameters (i.e., a, b, c) are assumed to be independent from user parameters θ. This assumption, for example, states that the difficulty of items is independent of users.

Independence among Items Item parameters are independent between items. This means, for example, that the model omits such a case in which some items cannot be answered correctly without knowing the answer of the other items.

4.4.2 Joint Maximum Likelihood Estimation

First, we introduce the joint maximum likelihood estimation (**JMLE**). This is one of the most simple estimation methods. We simply try to maximize the likelihood of a dataset by minimizing the negative log likelihood.

We define the log likelihood of the IRT. We denote the vector of parameters using bold fonts: i.e., we define $\boldsymbol{a} \overset{\text{def}}{=} \left(a_1, \ldots, a_{|V|}\right)^\top$, $\boldsymbol{b} \overset{\text{def}}{=} \left(b_1, \ldots, b_{|V|}\right)^\top$, $\boldsymbol{c} \overset{\text{def}}{=} \left(c_1, \ldots, c_{|V|}\right)^\top$, and $\boldsymbol{\theta} \overset{\text{def}}{=} \left(\theta_1, \ldots, \theta_{|U|}\right)^\top$. Then, the log likelihood of the model (4.3) of a dataset $\mathcal{D} \overset{\text{def}}{=} \{(y_1, u_1, v_1), \ldots, (y_N, u_N, v_N)\}$ given all the parameters can be written as follows. We assume all the data in the data $\mathcal{D}$ arise independently.

$$l_{IRT}\left(\mathcal{D}|\boldsymbol{\theta}, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}\right) \overset{\text{def}}{=} \prod_{i=1}^{N} P\left(1|u_i, v_i\right)^{y_i} P\left(0|u_i, v_i\right)^{(1-y_i)}. \tag{4.10}$$

We want to estimate the parameter vectors that maximize (4.10). To this end, we take the negative of the log of (4.10) and the negative log likelihood. For simple notation, we define $\sigma_{u,v}$ as the probability of taking 1 in (4.3) given a user u and a word v, i.e., $\sigma_{u,v} \overset{\text{def}}{=} P\left(1|u, v\right)$. Then, the negative log likelihood can be expressed as follows:

$$nll_{IRT}\left(\mathcal{D}|\boldsymbol{\theta}, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}\right) \overset{\text{def}}{=} -\sum_{i=1}^{N} \left(y_i \log \sigma_{u_i,v_i} + (1 - y_i) \log\left(1 - \sigma_{u_i,v_i}\right)\right). \tag{4.11}$$

We want to analyze the convexity of (4.11) because, if we can find (4.11) to be convex in some special cases, we can simply yield to sophisticated convex optimization methods. (4.11) is twice differentiable in terms of all the parameters. A twice differentiable function is convex if and only if its *Hessian* matrix is positive semi-definite (Section 3.1.4 of Boyd and Vandenberghe (2004)). Thus, we can determine the convexity of (4.11) by examining its Hessian matrix.

In Table 4.1, we summarize the convexity of (4.11). We can see that (4.11) is convex only for the Rasch model, or the one-parameter model. We show this by examining the Hessian matrix of (4.11).

By using the independence assumptions of §4.4.1, we can deduce the following equations, which are elements of the Hessian matrix of (4.11).

Independence of Users $\forall u, u' \in U, u \neq u'; \frac{\partial^2 nll_{IRT}}{\partial \theta_u \theta_{u'}} = 0$

Table. 4.1. Model specification and convexity of (4.11). Specification holds for $\forall v \in V$.

	Specification	Convexity of (4.11)
Rasch (one-parameter)	$a_v = 1, c_v = 0$	Convex
Two-parameter	$c_v = 0$	Non-convex
Three-parameter		Non-convex

Independence between Item Parameters and User Parameters $\forall u \in U, \forall v \in V$; $\frac{\partial^2 nll_{IRT}}{\partial \theta_u a_v} = \frac{\partial^2 nll_{IRT}}{\partial \theta_u b_v} = \frac{\partial^2 nll_{IRT}}{\partial \theta_u b_v} = 0$

Independence among Items $\frac{\partial^2 nll_{IRT}}{\partial a_v a_{v'}} = \frac{\partial^2 nll_{IRT}}{\partial b_v b_{v'}} = \frac{\partial^2 nll_{IRT}}{\partial c_v c_{v'}} = 0; \forall v, v' \in V, v \neq v'$

The other elements in the Hessian matrix of (4.11) can be written as follows. Note that we denote the set of all the indexes of the users that appear with a word v in a dataset $\mathcal{D}$ by $U_{\mathcal{D}}(v)$, i.e., $U_{\mathcal{D}}(v) = \{u' | (y', u', v') \in \mathcal{D}; v' = v\}$ We also denote the set of all the indexes of the words that appear with a user u in a dataset $\mathcal{D}$ by $V_{\mathcal{D}}(u)$, i.e., $V_{\mathcal{D}}(u) = \{v' | (y', u', v') \in \mathcal{D}; u' = u\}$.

$$\begin{aligned}
\frac{\partial^2 nll_{IRT}}{\partial \theta_u^2} &= -\sum_{v \in V_{\mathcal{D}}(u)} a_v^2 \frac{\sigma_{u,v} - c_v}{(1-c_v)^2} \frac{(1-\sigma_{u,v})}{\sigma_{u,v}} \frac{y_{u,v} c_v - \sigma_{u,v}^2}{\sigma_{u,v}} \\
\frac{\partial^2 nll_{IRT}}{\partial a_v^2} &= -\sum_{u \in U_{\mathcal{D}}(v)} (\theta_u - b_v)^2 \frac{\sigma_{u,v} - c_v}{(1-c_v)^2} \frac{(1-\sigma_{u,v})}{\sigma_{u,v}} \frac{y_{u,v} c_v - \sigma_{u,v}^2}{\sigma_{u,v}} \\
\frac{\partial^2 nll_{IRT}}{\partial b_v^2} &= -\sum_{u \in U_{\mathcal{D}}(v)} a_v^2 \frac{\sigma_{u,v} - c_v}{(1-c_v)^2} \frac{(1-\sigma_{u,v})}{\sigma_{u,v}} \frac{y_{u,v} c_v - \sigma_{u,v}^2}{\sigma_{u,v}} \\
\frac{\partial^2 nll_{IRT}}{\partial a_v b_v} &= \frac{1}{1-c_v} \sum_{u \in U_{\mathcal{D}}(v)} (\sigma_{u,v} - c_v) \left(1 - a_v (\theta_u - b_v) \frac{(1-\sigma_{u,v})}{1-c_v} \left(\frac{y_{u,v}}{\sigma_{u,v}} - 1\right)\right. \\
&\quad \left. + \frac{a_v}{1-c_v} (\theta_u - b_v) \frac{(1-\sigma_{u,v})}{\sigma_{u,v}} \left(\frac{y_{u,v} c_v}{\sigma_{u,v}} - y_{u,v}\right)\right). \qquad (4.12)
\end{aligned}$$

Convexity of Rasch model

First, we see the convexity of the negative log likelihood of the one-parameter model, or the Rasch model under the independence assumption §4.4.1. In this case, $c_v = 0, a_v = 1$ hold true for all the words $v \in V$. By substituting $c_v = 0, a_v = 1$ in (4.12), we have the following formulae. Note that, as a_v is set to be a constant, all the derivatives by a_v are 0; thus, we do not need to take them into account in the Rasch model.

$$\frac{\partial^2 nll_{IRT}}{\partial \theta_u^2} = \sum_{v \in V_{\mathcal{D}}(u)} (1-\sigma_{u,v})\,\sigma_{u,v} \geq 0$$
$$\frac{\partial^2 nll_{IRT}}{\partial b_v^2} = \sum_{u \in U_{\mathcal{D}}(v)} (1-\sigma_{u,v})\,\sigma_{u,v} \geq 0. \tag{4.13}$$

Here, the derivatives are non-negative since $\sigma_{u,v}$ is a probability; thus, the following holds true.

$$0 \leq \sigma_{u,v} \leq 1$$
$$0 \leq 1-\sigma_{u,v} \leq 1$$
$$(1-\sigma_{u,v})\sigma_{u,v} \geq 0.$$

To see the semi-definiteness of the Hessian matrix of (4.11), we denote the Hessian matrix of (4.11) by $\nabla^2{}_{nll_{IRT}}$. This matrix has the size of the square of $|U|+3|V|$. We also define a vector of $|U|+3|V|$ dimensions, $\mathbf{z}$ as follows:

$$\mathbf{z} = \left(z_{\theta_1}, \ldots, z_{\theta_{|U|}}, z_{a_1}, \ldots, z_{a_{|V|}}, z_{b_1}, \ldots z_{b_{|V|}}, z_{c_1}, \ldots z_{c_{|V|}}\right)^\top \in \mathbb{R}^{|U|+3|V|}. \tag{4.14}$$

Finally, we can show the semi-definiteness of the Hessian matrix of (4.11) as follows. By using the inequalities in (4.13), we can see that the following holds true.

$$\mathbf{z}^\top \nabla^2{}_{nll_{IRT}} \mathbf{z} = \sum_{i=1}^{N} \left(z_{\theta_{u_i}}^2 \frac{\partial^2 nll_{IRT}}{\partial \theta_{u_i}^2} + z_{b_{v_i}}^2 \frac{\partial^2 nll_{IRT}}{\partial b_{v_i}^2} \right) \geq 0. \tag{4.15}$$

Even if the assumption shown in §4.4.1 does not hold, we can see that the negative log likelihood of the Rasch model is convex from the fact that the Rasch model is a special case of a binary logistic regression as we saw in §4.3, and the negative log likelihood of a binary logistic regression is convex. By using the convexity of a binary logistic regression, we can also see that the negative log likelihood of the shared difficulty model and LLTM explained in §4.3 is also convex.

We then see the non-convexity of the two-parameter model. As the three-parameter model contains the two-parameter model as a special case when c_v is set to be $c_v = 0$, it is sufficient to show the non-convexity of the two-parameter model to show the non-convexity of the three parameter model.

Non-Convexity of Two-Parameter and Three-Parameter Models

We then see the non-convexity of the two-parameter and the three-parameter models. As the two-parameter model is a special case of the three-parameter model, it is sufficient to see that the two-parameter model is non-convex. In the two-parameter model, $c_v = 0$ always holds true for all $v \in V$.

Unlike proofs of convexity, to see non-convexity, we only have to show one counter-example that the Hessian of (4.11) can be negative. To this end, we write the second derivatives of the two parameter models as follows:

$$\begin{aligned}
\frac{\partial^2 nll_{IRT}}{\partial \theta_u^2} &= \sum_{v \in V_{\mathcal{D}}(u)} a_v^2 \left(1 - \sigma_{u,v}\right) \sigma_{u,v} \geq 0 \\
\frac{\partial^2 nll_{IRT}}{\partial a_v^2} &= \sum_{u \in U_{\mathcal{D}}(v)} \left(\theta_u - b_v\right)^2 \left(1 - \sigma_{u,v}\right) \sigma_{u,v} \geq 0 \\
\frac{\partial^2 nll_{IRT}}{\partial b_v^2} &= \sum_{u \in U_{\mathcal{D}}(v)} a_v^2 \left(1 - \sigma_{u,v}\right) \sigma_{u,v} \geq 0 \\
\frac{\partial^2 nll_{IRT}}{\partial a_v b_v} &= \sum_{u \in U_{\mathcal{D}}(v)} \sigma_{u,v} \Biggl(1 - a_v \left(\theta_u - b_v\right) \left(1 - \sigma_{u,v}\right) \left(\frac{y_{u,v}}{\sigma_{u,v}} - 1\right) \\
&\quad + a_v \left(\theta_u - b_v\right) \frac{\left(1 - \sigma_{u,v}\right)}{\sigma_{u,v}} \left(-y_{u,v}\right)\Biggr) . \qquad (4.16)
\end{aligned}$$

In (4.16), we can see that the sign of the last equation can not be determined because it is the derivative of two different types of parameters; a_v and b_v. In contrast, we can see that the first three pure equations are non-negative. The last equation of (4.16) arises because of relaxing the constraint $a_v = 1$, which holds true in the one-parameter model, or the Rasch model.

To see the sign of the Hessian matrix of (4.11) in the two-parameter model, we calculate its quadratic form as follows by using (4.16).

$$\begin{aligned}
&\mathbf{z}^\top \nabla^2{}_{nll_{IRT}} \mathbf{z} \\
&= \sum_{i=1}^{N} \left(z_{\theta_{u_i}}^2 \frac{\partial^2 nll_{IRT}}{\partial \theta_{u_i}^2} + z_{a_{v_i}}^2 \frac{\partial^2 nll_{IRT}}{\partial a_{v_i}^2} + z_{b_{v_i}}^2 \frac{\partial^2 nll_{IRT}}{\partial b_{v_i}^2} + 2 z_{a_{v_i}} z_{b_{v_i}} \frac{\partial^2 nll_{IRT}}{\partial a_{v_i} b_{v_i}} \right) . (4.17)
\end{aligned}$$

In order to see that (4.17) can be negative, we consider a counter-example when $\theta_u = b_v$ holds true for simplicity. Again, note that we only have to pose a counter example to see the non-convexity. When $\theta_u = b_v$ holds true for all users $u \in U$ and words $v \in V$ in the dataset $\mathcal{D}$ in the two-parameter models, in which $c_v = 0$ holds true, from (4.3), we can obtain $\sigma_{u,v} = \frac{1}{2}$. Substituting $\theta_u = b_v$ and $\sigma_{u,v} = \frac{1}{2}$ into (4.17), we can obtain the following.

$$\begin{aligned}
\frac{\partial^2 nll_{IRT}}{\partial \theta_u^2} &= \sum_{v \in V_{\mathcal{D}}(u)} \frac{1}{4} a_v^2 \geq 0 \\
\frac{\partial^2 nll_{IRT}}{\partial a_v^2} &= 0 \\
\frac{\partial^2 nll_{IRT}}{\partial b_v^2} &= \sum_{u \in U_{\mathcal{D}}(v)} \frac{1}{4} a_v^2 \geq 0 \\
\frac{\partial^2 nll_{IRT}}{\partial a_v b_v} &= \sum_{u \in U_{\mathcal{D}}(v)} \frac{1}{2} \geq 0
\end{aligned} \tag{4.18}$$

From (4.18), all the equations including the last equation are non-negative when $\theta_u = b_v$ holds true. Substituting (4.18) into (4.17), the inner part of the sum of (4.17) can be written as follows:

$$\begin{aligned}
& z_{\theta_u}^2 \frac{\partial^2 nll_{IRT}}{\partial \theta_u^2} + z_{b_v}^2 \frac{\partial^2 nll_{IRT}}{\partial b_v^2} + 2 z_{a_v} z_{b_v} \frac{\partial^2 nll_{IRT}}{\partial a_v b_v} \\
&= z_{\theta_u}^2 \left(\sum_{v \in V_{\mathcal{D}}(u)} \frac{1}{4} a_v^2 \right) + z_{b_v}^2 \left(\sum_{u \in U_{\mathcal{D}}(v)} \frac{1}{4} a_v^2 \right) + 2 z_{a_v} z_{b_v} \left(\sum_{u \in U_{\mathcal{D}}(v)} \frac{1}{2} \right).
\end{aligned} \tag{4.19}$$

In (4.19), all the terms within the brackets are non-negative, and we can choose all zs arbitrarily. Thus, if we set $z_{\theta_u} = z_{b_v} = 1$ and z_{a_v} to be a sufficiently small negative value, (4.19) can be negative. Thus, the Hessian matrix of (4.11) is not positive semi-definite in the two-parameter model; thus, the joint negative log likelihood of the two-parameter and three-parameter models are non-convex. This is contrasting compared with the one-parameter model, or the Rasch model, whose joint negative log likelihood is convex.

4.4.3 Marginal Maximum Likelihood Estimation

In the previous subsection, we surveyed the joint maximum likelihood estimation (JMLE) and showed that it is non-convex in the two-parameter and three-parameter models. In addition to non-convexity, it is known that there is another problem with JMLE when it is used for estimating parameters of the two-parameter and three-parameter models: *consistency* (Baker and Kim, 2004). It is known that the IRT parameters estimated using JMLE are not consistent estimators. They are inconsistent as the size of the test increases. The reason is because the two types of parameters, namely the item parameters $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$ and the user parameters $\boldsymbol{\theta}$, are jointly estimated in JMLE (p. 157, Baker and Kim (2004)). Because of this shortcoming, JMLE is seldom used as a parameter estimation method of the two and three-parameter models, while it is common to use it for the Rasch model.

To tackle the consistency problem, methods that can be used to avoid joint estimation were proposed such as Bock and Liberman (1970); Bock and Aitkin (1981). The key idea of Bock and Liberman (1970) is to, first marginalize out the users' ability parameters in the likelihood and maximize the marginalized likelihood. Thus, this method is called the *marginal maximum likelihood estimation* (**MMLE**). However, Bock and Liberman (1970) has another problem in that the Hessian matrix involving its estimation is too large to fit in the memory of computers. Bock and Aitkin (1981) solve this memory problem by using the expectation-maximization (EM) algorithm. Bock and Liberman (1970); Bock and Aitkin (1981) are now standard methods for estimating the parameters of the two- and three-parameter models.

In this section, following Baker and Kim (2004), we briefly introduce Bock and Liberman (1970); Bock and Aitkin (1981) because they are current standard of the parameter estimation of the two and three parameter models.

Bock & Lieberman's method

First, $\boldsymbol{y}_u$ denotes the vector of user u's responses for all the words that he/she responded to: i.e., $\mathbf{y}_u \overset{\text{def}}{=} \{y' | \forall u'; (y', u', v') \in \mathcal{D}, u' = u\}$. Then, the probability of observing $\boldsymbol{y}_u$ given all the other parameters $\theta_u, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$ can be written as follows:

$$P\left(\mathbf{y}_u | \theta_u, \mathbf{a}, \mathbf{b}, \mathbf{c}\right) = \prod_{v \in V_{\mathcal{D}(u)}} P\left(1 | u, v; \theta_u, \mathbf{a}, \mathbf{b}, \mathbf{c}\right)^{y_{u,v}} P\left(0 | u, v; \theta_u, \mathbf{a}, \mathbf{b}, \mathbf{c}\right)^{1 - y_{u,v}}. \tag{4.20}$$

Then, we marginalize (4.20) by θ_u since we want to avoid maximizing item parameters $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$ and user u's ability parameter θ_u since it is not statistically consistent as we explained.

$$P\left(\mathbf{y}_u | \mathbf{a}, \mathbf{b}, \mathbf{c}, \boldsymbol{\tau}\right) = \int_{-\infty}^{\infty} P\left(\mathbf{y}_u | \theta_u, \mathbf{a}, \mathbf{b}, \mathbf{c}\right) g\left(\theta | \boldsymbol{\tau}\right) d\theta. \tag{4.21}$$

In (4.21), $g(\theta_u|\boldsymbol{\tau})$ is a prior distribution of θ_u, and $\boldsymbol{\tau}$ is the hyper parameters of θ_u. It is assumed that the same prior distribution $g(\theta_u|\boldsymbol{\tau})$ is applicable to all the users.

Then, the *marginalized likelihood* can be written as follows:

$$l_{marg}\left(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}\right) \overset{\text{def}}{=} \prod_{u \in U} P\left(\mathbf{y}_u | \mathbf{a}, \mathbf{b}, \mathbf{c}, \boldsymbol{\tau}\right). \tag{4.22}$$

By taking the negative log of (4.22), we obtain the marginalized negative log likelihood.

$$nll_{marg}\left(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}\right) \overset{\text{def}}{=} -\sum_{u \in U} \log P\left(\mathbf{y}_u | \mathbf{a}, \mathbf{b}, \mathbf{c}, \boldsymbol{\tau}\right). \tag{4.23}$$

The item parameters $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$ can be consistently estimated by minimizing (4.23) because the user parameter θ_u is marginalized out. To minimize (4.23), we derive the derivatives of (4.23) in terms of $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$.

$$\frac{\partial nll_{marg}}{\partial a_u} = -\left(1-c_v\right) \sum_{u \in U_{\mathcal{D}}(v)} \int_{-\infty}^{\infty} \left(y_{u,v} - P\left(1|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right)\right)$$

$$\gamma_{u,v}\left(\theta_u - b_v\right) P\left(\theta_u|\mathbf{y}_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) d\theta_u \tag{4.24}$$

$$\frac{\partial nll_{marg}}{\partial b_v} = a_v\left(1-c_v\right) \sum_{u \in U_{\mathcal{D}}(v)} \int_{-\infty}^{\infty} \left(y_{u,v} - P\left(1|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right)\right)$$

$$\gamma_{u,v} P\left(\theta_u|\mathbf{y}_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) d\theta_u \tag{4.25}$$

$$\frac{\partial nll_{marg}}{\partial c_v} = -\left(1-c_v\right)^{-1} \sum_{u \in U_{\mathcal{D}}(v)} \int_{-\infty}^{\infty} \frac{y_{u,v} - P\left(1|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right)}{P\left(1|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right)}$$

$$P\left(\theta_u|\mathbf{y}_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) d\theta_u. \tag{4.26}$$

Here, $\gamma_{u,v}$ and $P\left(\theta_u|\mathbf{y}_u,\mathbf{a},\mathbf{b},\mathbf{c}\right)$ can be written and defined, respectively, as follows:

$$P\left(\theta_u|\mathbf{y}_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) = \frac{P\left(\mathbf{y}_u|\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) g\left(\theta|\tau\right)}{\int_{-\infty}^{\infty} P\left(\mathbf{y}_u|\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) g\left(\theta|\tau\right) d\theta} \tag{4.27}$$

$$\gamma_{u,v} \overset{\text{def}}{=} \frac{P^*\left(1|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) P^*\left(0|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right)}{P\left(1|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) P\left(0|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right)}. \tag{4.28}$$

Here, $P^*\left(1|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right)$ and $P^*\left(0|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right)$ are the probabilities of responding to 1 and 0, respectively, when $\boldsymbol{a}, \boldsymbol{b}$ are applied to the two-parameter models, i.e., $\boldsymbol{c} = \mathbf{0}$. They can be written as follows:

$$P^*\left(1|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) \overset{\text{def}}{=} \frac{P\left(1|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) - c_i}{1-c_i} \tag{4.29}$$

$$P^*\left(0|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) = 1 - P^*\left(1|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) = \frac{1 - P\left(1|u,v;\theta_u,\mathbf{a},\mathbf{b},\mathbf{c}\right)}{1-c_i}. \tag{4.30}$$

The integrals in (4.24), (4.25), and (4.26) makes it difficult to obtain an analytic formulation. Therefore, *numerical integration* is often applied. We introduce the case in which one of the most simple numerical integration methods, *rectangle rule*, is applied to approximate the integrals in the first derivatives.

Suppose that we take Q points, $\{\tilde{\theta}_1|q \in \{1,\ldots,Q\}\}$, to approximate the integrals. We denote the value of $g(\theta|\boldsymbol{\tau})$ in the neighborhood of $\theta = \tilde{\theta}_q$ by $A(\tilde{\theta}_q)$. Then, the first derivatives shown in

(4.24), (4.25), and (4.26) can be approximated by using the rectangle rule, as follows:

$$\frac{\partial nll_{marg}}{\partial a_u} \approx -\left(1-c_v\right) \sum_{u \in U_{\mathcal{D}}(v)} \sum_{q=1}^{Q} \left(y_{u,v} - P\left(1|u,v;\tilde{\theta}_q,\mathbf{a},\mathbf{b},\mathbf{c}\right)\right)$$
$$\gamma_{u,v}\left(\tilde{\theta}_q - b_v\right) P\left(\tilde{\theta}_q|\mathbf{y}_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) \tag{4.31}$$

$$\frac{\partial nll_{marg}}{\partial b_v} \approx a_v\left(1-c_v\right) \sum_{u \in U_{\mathcal{D}}(v)} \sum_{q=1}^{Q} \left(y_{u,v} - P\left(1|u,v;\tilde{\theta}_q,\mathbf{a},\mathbf{b},\mathbf{c}\right)\right)$$
$$\gamma_{u,v} P\left(\tilde{\theta}_q|\mathbf{y}_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) \tag{4.32}$$

$$\frac{\partial nll_{marg}}{\partial c_v} \approx -\left(1-c_v\right)^{-1} \sum_{u \in U_{\mathcal{D}}(v)} \sum_{q=1}^{Q} \frac{y_{u,v} - P\left(1|u,v;\tilde{\theta}_q,\mathbf{a},\mathbf{b},\mathbf{c}\right)}{P\left(1|u,v;\tilde{\theta}_q,\mathbf{a},\mathbf{b},\mathbf{c}\right)}$$
$$P\left(\tilde{\theta}_q|\mathbf{y}_u,\mathbf{a},\mathbf{b},\mathbf{c}\right). \tag{4.33}$$

Here, $P\left(\tilde{\theta}_q|\mathbf{y}_u,\mathbf{a},\mathbf{b},\mathbf{c}\right)$ can be expressed as follows:

$$P\left(\tilde{\theta}_q|\mathbf{y}_u,\mathbf{a},\mathbf{b},\mathbf{c}\right) = \frac{\prod_{v \in U_{\mathcal{D}}(u)} P\left(1|u,v;\tilde{\theta}_q,\mathbf{a},\mathbf{b},\mathbf{c}\right)^{y_{u,v}} P\left(0|u,v;\tilde{\theta}_q,\mathbf{a},\mathbf{b},\mathbf{c}\right)^{1-y_{u,v}} A\left(\tilde{\theta}_q\right)}{\sum_{q=1}^{Q} \prod_{v \in U_{\mathcal{D}}(u)} P\left(1|u,v;\tilde{\theta}'_q,\mathbf{a},\mathbf{b},\mathbf{c}\right)^{y_{u,v}} P\left(0|u,v;\tilde{\theta}'_q,\mathbf{a},\mathbf{b},\mathbf{c}\right)^{1-y_{u,v}} A\left(\tilde{\theta'}_q\right)}. \tag{4.34}$$

Finally, we can calculate the first derivatives by substituting (4.34) into (4.31), (4.32), and (4.33) to minimize the negative log likelihood shown in (4.23).

The concrete methodology to minimize the negative log likelihood, (4.23), varies by implementations. A straight-forward way to minimize the negative log likelihood is to, first obtain the second derivatives by using numerical differentiation, then apply the Newton-Raphson algorithm to minimize (4.23). Baker and Kim (2004) states that the original Bock and Liberman (1970) used this methodology. This straight-forward methodology, however, has a notable problem in that the Hessian matrix used in the Newton-Raphson algorithm becomes too large to fit in the memory of computers because it has the size of the square of $|U| + 3|V|$. To tackle this problem, Bock and Aitkin (1981), the method that we introduce in the next subsection, was proposed.

However, modern computers may not be affected by the memory problem. The reason of this is that

- first, modern computers are equipped with considerably large memory compared with computers at the time that Bock and Liberman (1970) was proposed [Here you are saying that the autors themselves were proposed].
- Second, the Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) method was proposed Liu and Nocedal (1989) since then. This quasi-Newton method requires only

linear-size memory unlike the original Newton-Raphson algorithm, which requires square-size memory.

However, as Bock and Aitkin (1981) has become a standard methodology for estimating the parameters of the two- and three-parameter IRT models, it is still used widely today. Also, we can use a more sophisticated numerical integration method, for example, the *Gauss-Hermite quadrature*, which some modern implementations are equipped with. Again, note that concrete algorithms used vary by implementations. In the latter part of this chapter, we summarize what method is used by taking some well known implementations as examples.

Finally, we want to remark on convexity, which we focused on when we explained JMLE.

Bock & Aitkin's method

Here, following Baker and Kim (2004), we introduce the EM-algorithm method used in Bock and Aitkin (1981).

We define $\xi_{v,q}$ and $\xi'_{v,q}$ as follows:

$$\xi_{v,q} \overset{\text{def}}{=} \sum_{u \in U_{\mathcal{D}}(v)} y_{u,v} P\left(\tilde{\theta}_q | \mathbf{y}_u, \mathbf{a}, \mathbf{b}, \mathbf{c}\right)$$

$$= \sum_{u \in U_{\mathcal{D}}(v)} \frac{y_{u,v} \prod_{v \in U_{\mathcal{D}}(u)} P\left(1|u,v;\tilde{\theta}_q, \mathbf{a}, \mathbf{b}, \mathbf{c}\right)^{y_{u,v}} P\left(0|u,v;\tilde{\theta}_q, \mathbf{a}, \mathbf{b}, \mathbf{c}\right)^{1-y_{u,v}} A\left(\tilde{\theta}_q\right)}{\sum_{q=1}^{Q} \prod_{v \in U_{\mathcal{D}}(u)} P\left(1|u,v;\tilde{\theta}'_q, \mathbf{a}, \mathbf{b}, \mathbf{c}\right)^{y_{u,v}} P\left(0|u,v;\tilde{\theta}'_q, \mathbf{a}, \mathbf{b}, \mathbf{c}\right)^{1-y_{u,v}} A\left(\tilde{\theta}'_q\right)} \tag{4.35}$$

$$\tag{4.36}$$

$$\xi'_{v,q} \overset{\text{def}}{=} \sum_{u \in U_{\mathcal{D}}(v)} P\left(\tilde{\theta}_q | \mathbf{y}_u, \mathbf{a}, \mathbf{b}, \mathbf{c}\right)$$

$$= - \sum_{u \in U_{\mathcal{D}}(v)} \frac{\prod_{v \in U_{\mathcal{D}}(u)} P\left(1|u,v;\tilde{\theta}_q, \mathbf{a}, \mathbf{b}, \mathbf{c}\right)^{y_{u,v}} P\left(0|u,v;\tilde{\theta}_q, \mathbf{a}, \mathbf{b}, \mathbf{c}\right)^{1-y_{u,v}} A\left(\tilde{\theta}_q\right)}{\sum_{q=1}^{Q} \prod_{v \in U_{\mathcal{D}}(u)} P\left(1|u,v;\tilde{\theta}'_q, \mathbf{a}, \mathbf{b}, \mathbf{c}\right)^{y_{u,v}} P\left(0|u,v;\tilde{\theta}'_q, \mathbf{a}, \mathbf{b}, \mathbf{c}\right)^{1-y_{u,v}} A\left(\tilde{\theta}'_q\right)} \tag{4.37}$$

An EM-algorithm searches a local minimum by iterating an **E-step** (Expectation step) and **M-step** (Maximization step). (4.35) and (4.37) correspond to the E-step. The M-step can be written in (4.38), (4.39), and (4.40). These M-step formulae can be derived by reforming (4.31), (4.32), and (4.33) by using (4.35) and (4.37).

$$a_v{:}-(1-c_v)\sum_{q=1}^{Q}\left(\tilde{\theta}_q-b_v\right)\left(\xi_{u,v}-\xi'_{u,v}P\left(1|u,v;\tilde{\theta}_q,\mathbf{a},\mathbf{b},\mathbf{c}\right)\right)\gamma_{u,v}=0, \tag{4.38}$$

$$b_v{:}a_v\,(1-c_v)\sum_{q=1}^{Q}\left(\xi_{u,v}-\xi'_{u,v}P\left(1|u,v;\tilde{\theta}_q,\mathbf{a},\mathbf{b},\mathbf{c}\right)\right)\gamma_{u,v}=0, \tag{4.39}$$

$$c_v{:}(1-c_v)^{-1}\sum_{q=1}^{Q}\frac{\left(\xi_{u,v}-\xi'_{u,v}P\left(1|u,v;\tilde{\theta}_q,\mathbf{a},\mathbf{b},\mathbf{c}\right)\right)}{P\left(1|u,v;\tilde{\theta}_q,\mathbf{a},\mathbf{b},\mathbf{c}\right)}=0. \tag{4.40}$$

4.4.4 Implementation

The parameter estimation methods in the two- and three-parameter IRT models differ by implementation. We introduce notable freely-available implementations.

Recently, the programming language **R** has been extensively used for statistical computing. Users of R can handle new statistical models by adding *packages* of R. There are mainly two packages that support the IRT: the “ltm” package and “irtoys” package. As each package is required to attach an instructional document in R, the parameter estimation methods that these packages use can also be found in the documents.

According to the documentation of the “ltm” package, it uses MMLE as the parameter estimation method. Unlike the simple “rectangle rule” that we expained, the “ltm” package uses the Gauss-Hermite quadrature for approximating the integrals in the first derivatives of the marginal log likelihood. To maximize the marginal likelihood, It uses a “hybrid algorithm” that uses the EM-algorithm by Bock and Aitkin (1981) for some initial iterations, then it switches to a quasi-Newton method.

While the “ltm” package is fully written in R, the “irtoys” package is designed to call external software for its parameter estimation. By default, “irtoys” attempts to call the IRT Command Language (ICL) (Hanson), which also supports the Bayesian modal estimation method, Mislevy (1986), another standard parameter estimation method for the two- and three-parameter IRT models. Note that both “irtoys” and ICL have not been updated for over five years.

4.5 Parameter Estimation Methods of Convex Models

This section introduces parameter estimation methods for convex models. The main parameter estimation algorithms are listed in Table 4.2.

The “Optimal” column shows whether the algorithm can find the global optimal solution of the objective function of a logistic regression. The “Batch/Online” column shows whether the

Table. 4.2. Parameter estimation methods for logistic regression.

	Optimal	Batch/Online	General
L-BFGS Liu and Nocedal (1989)	$\surd$	Batch	✓
Trust Region Lin et al. (2008)	$\surd$	Batch	LR only
SGD Bishop (2006)		Online	LR only

algorithm is a batch algorithm or an online algorithm. The “General” column shows to which problem the method is applicable. If the “General” column is ✓, it means that the method is applicable to all continuous twice-differentiable convex functions. If the “General” column is “LR only”, the method is applicable only to the logistic regression.

The Limited-memory Broyden-Fletcher-Goldfarb-Shanno method (L-BFGS, Liu and Nocedal (1989), is a widely used method for estimating logistic regression parameters. It is a variant of the quasi-Newton method. It consumes a memory space only linear to the number of features while a naïve method consumes a memory space squared to the number of features.

Trust region, Lin et al. (2008), is another recently proposed batch method for optimizing binary logistic regression parameters. It can optimize the parameters faster than L-BFGS, but it is limited to logistic regression whereas L-BFGS is more general and more widely applicable. An implementation of Trust region is given in Fan et al. (2008) and this implementation was used for the evaluation described later.

4.5.1 Stochastic Gradient Descent (SGD)

While a batch algorithm requires all the training data to be given from the beginning to the end, an online algorithm can process every record of the data one by one. Stochasitic gradient descent (SGD) is an online algorithm and is derived from steepest gradient descent, a batch algorithm, by removing the summation over the data points. The steepest gradient descent for logistic regression is shown in (4.41), where $\nabla nll_i(\boldsymbol{w}) \overset{\mathrm{def}}{=} \left(y_i - \sigma\left(\boldsymbol{w}^{\mathrm{T}}\phi(v_i)\right)\right) \boldsymbol{\phi}_n$1. Here, $\boldsymbol{\phi}_i = \boldsymbol{\phi}(\boldsymbol{x}_i)$, where $\boldsymbol{x}_i$ is the $n \in \{1, \ldots, N\}$th click log among all N click logs.

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \sum_{i=1}^{N} \nabla E_i(\boldsymbol{w}^{(t)}) \tag{4.41}$$

(4.41) contains the summation of nll_i over all the data, so it is a batch algorithm. In this case, an online algorithm is easily derived by removing the summation over i from (4.41). The resulting online algorithm is SGD, the algorithm used in our system, and is given by

NOT

PUBLIC

Fig. 4.7. “Mean difficulties and 95% confidence intervals for the 14 levels”. Taken from Figure 2 of Beglar (2010).

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta_t \nabla nll_i(\mathbf{w}^{(t)}) \tag{4.42}$$

In (4.42), η_t is defined as $\eta_t = \frac{1}{\rho(t+t_0)}$, where ρ, t_0 are the parameters of SGD and η_t is defined in this way so that (4.42) converges (Novikoff, 1963). Note that η_t is usually a constant in (4.41).

NOT

PUBLIC

Fig. 4.8. “Conversion table of Rasch ability estimates and number of word families known”. Taken from Table 2 of Beglar (2010).

4.6 Rasch-based Validation of Vocabulary Size Test

The Vocabulary Size Test was later thoroughly validated using the Rasch model in Beglar (2010). David Beglar is also a co-author of the Vocabulary Size Test (Nation and Beglar, 2007). Beglar (2010) linked the Rasch model and the Vocabulary Size Test we explained in the previous chapter. In this section, we briefly explain what Beglar (2010) showed by using the Rasch model. Beglar (2010) tested 197 Japanese English users by using the Vocabulary Size Test.

We introduce the most important figures from Beglar (2010): Figure 4.7, and Figure 4.8. In this chapter, we explained how the Rasch model works and that it has the difficulty parameter. In 3, we saw that with most tests, including the Vocabulary Size Test, it is assumed that the words with low corpus frequency should be easy for users. Although this assumption is intuitively true, this assumption requires validation. Some low-ability users may know difficult words.

Figure 4.7 plots word frequency level versus the mean and variance of the difficulty parameter b_v of the words in each level. Remember that the Vocabulary Size Test samples 10 words from each word frequency level consisting of $1,000$ words. Thus, each circle in Figure 4.7 denotes the mean of the 10 words in each level, and the bar denotes the standard deviation of the 10 words of the level.

From Figure 4.7, we can see that the word frequency levels and the difficulty parameter of the Rasch model basically correlate, except the drop at $8,000$. Beglar (2010) hypothesized that this drop is due to the effect of *loanwords*. A loanword is a word that is imported to the users' native language from other languages. For example, the Japanese word "ko-hi-" is a loanword from English "coffee". However, as the purpose of Beglar (2010) did not lie in analyzing the data further regarding this issue, they did not go further to validating this hypothesis.

Beglar (2010) also presented the relation between the Rasch ability estimate and the *vocabulary size*, as shown in Figure 4.7. They also pointed out the superiority of the Rasch ability estimate against the calculated vocabulary size. The scoring of users by using the Rasch ability estimate is more robust than the scoring using vocabulary size. Beglar (2010) listed "a number of advantages" of the Rasch ability-based scoring over the vocabulary-size-based scoring: "missing responses do not present a problem, idiosyncratic answering patterns by test-takers and idiosyncratic item performance can be identified though the inspection of person and item fit indices, the degree of invariance of different test forms can be readily identified, and through the use of carefully selected anchor items, a multitude of test forms can be used to place test-takers on a single continuum of breadth of written receptive vocabulary knowledge, provided that the items fit the Rasch model and that the property of item invariance holds."

In summary, it can be said that Beglar (2010) is positive on using the Rasch ability parameter instead of the vocabulary size when the former is obtainable. As we noted in the previous chapter, the *vocabulary size* in the vocabulary tests seems to be useful as an indicator of English vocabulary proficiency, rather than as an indicator of the actual number of words that a user knows.

4.7 Chapter Summary

In this chapter, we surveyed the IRT, which consists of three models: the Rasch model (the one-parameter model), the two-parameter model, and the three-parameter model. The containment relationship of these models are shown in Figure 4.1. We argued that only the Rasch model and its extension, the shared difficulty model, are convex. The two- and three-parameter models are not convex.

Chapter 5

User-specific Word Difficulty

This chapter is not public for journal publication and copyright issues. References to sections, figures and equations of this chapter in the other chapters are shown in "??".

Chapter 6

Application of Vocabulary Prediction to Reading Support

6.1 Introduction

English is now so widespread that, in non-English speaking countries, office workers who are not specializing in English need to read English documents during their office work. Furthermore, more and more users are now browsing Web pages, using English word glossing systems, which eliminate the need to consult a dictionary every time an unknown word is encountered. A glossing system presents the user with the meaning of an unknown word in a pop-up window when the user clicks on or mouses over it. An example is “pop jisyo”, shown in Figure 2.5. The background sentences that begin with “A group of Han Chinese” are sentences on a Web page that we assume the user is reading. Suppose that the user encounters the word “paramilitary” and does not know its meaning. If the user mouses over the word, a pop-up window opens and the meaning of the word “paramilitary” is displayed in it.

While useful, existing English glossing systems do not utilize the user’s vocabulary. They waste valuable data about the user’s vocabulary, which could be accumulated. Instead, we may harness collective intelligence by utilizing the accumulated word click logs from many users. We are developing a system that can accumulate this kind of knowledge and make full use of it to help users to read English Web pages by predicting the words that the user does not know and displaying their meanings [*1]. Our aim is to combine an English word glossing system with word difficulty prediction in an integrated and intelligent user interface. Existing research on word difficulty relies heavily on corpus frequency. However, corpus frequency may not reflect the

[*1] `http://www.socialdict.com/`

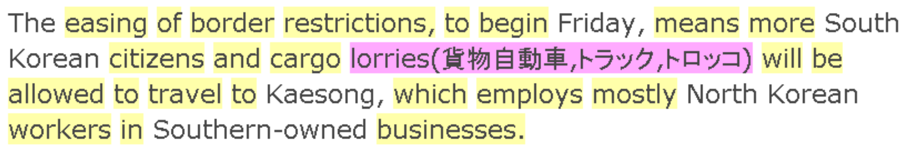
difficulty that users actually feel: some words occur frequently in corpora, but actually may not be known to the users, while some words are rarely seen in corpora, but actually may be known to the users. By accumulating the word click logs from many users and uncovering the "collective intelligence" beneath the logs, we can get the collective intelligence to adjust the weight for each word to reflect the difficulty that users actually feel; even if some words occur frequently in corpora, those words are regarded as difficult if many users frequently click them to see their glosses. Even if some words are rarely seen in corpora, those words are regarded as easy if many users click those words to hide their glosses. Thus, we can regard our system as using the collective intelligence embedded in the click log to correct the corpus-frequency-based word difficulty.

The main contribution of this chapter is a mechanism for predicting words unknown to a user. We use logistic regression to estimate which words on the Web page are unknown to the user because it is widely used to test language ability in "item response theory" (IRT). This theory lets one estimate both a user's language ability θ and a word's difficulty b at the same time. To train the logistic regression faster, we chose stochastic gradient descent (SGD) from among many parameter optimization methods because it is an online algorithm. An online algorithm can perform training faster than a batch method because it accesses a datum in the data set only once whereas a batch method accesses the data many times.

The example in Figure 6.1 shows how the system works when a reader, being an English user as well, tries to read a Web page. The yellow words are predicted to be known to this reader and the red word, namely "lorries" in this example, is the word predicted to be unknown to the user. Thus, this red word is "glossed", that is, a gloss is attached to it. The system makes this prediction when the reader loads the Web page so that, from the reader's viewpoint, the Web page seems to come with the gloss in the first place. The reader can also mispredictions by clicking words: when the reader clicks a yellow (i.e., predicted to be known) word, the system is informed that the reader actually does not know the word. When the reader clicks a red (i.e., predicted to be unknown), word, the system is informed that the reader actually "knows" the word. These corrections are instantly dispatched to the system from the reader's Web browser by an Ajax script, so that the system can learn and reflect the correction in its prediction when the reader goes to another Web page.

Another contribution of this chapter is that we evaluated our system to see whether it can reduce the number of clicks and the number of unreadable documents. The t-test was performed for both evaluations and the results show that the system worked significantly well for these evaluations.

In summary, our system is characterized as follows.



The easing of border restrictions, to begin Friday, means more South Korean citizens and cargo lorries(貨物自動車,トラック,トロッコ) will be allowed to travel to Kaesong, which employs mostly North Korean workers in Southern-owned businesses.

Fig. 6.1. Example of word glossing in our system.

1. Accumulates knowledge about the vocabulary of each user and fully utilizes it
2. Has a machine-learning-based mechanism for predicting words unknown to a user

First, we introduce related work. Second, we introduce the design and implementation issues of our system. We then explain the relationships among IRT, logistic regression, and the Rasch model, which is a special case of IRT and also a special case of logistic regression, which our system uses. We then explain the methods for estimating the parameters of logistic regression and the derivation of SGD. We then describe an experimental evaluation of our system. Finally, we conclude this chapter by briefly summarizing the main points and mentioning future work.

Note that throughout this chapter we use the term "unfamiliar words" to denote words unknown to a user. Although we know that word familiarity is, strictly speaking, a different concept from simple word knowledge, since word familiarity is not the focus of this chapter and the term "unknown words" has another meaning, we use "unfamiliar words" for simplicity.

6.2 System Overview

This section explains the operation of our system and implementation issues. Since we accumulate and utilize the click logs, we implemented our system as a Web application.

6.2.1 System Architecture

The way our system operates is schematically illustrated in Figure 6.2. It is a glossing system that uses "CGI-proxy" and predicts words unknown to the user by machine learning. In use, it operates as follows.

(0) The user passes user identifier u to the system. Note that a user identifier is not necessary in previous systems because they do not perform user adaptation.

(1) The user passes document identifier $d \in URL$ to the system.

(2) The system finds the Web server hosting the target document by following d.

(3) The system tries to retrieve document D specified by d from the Web server. If the system

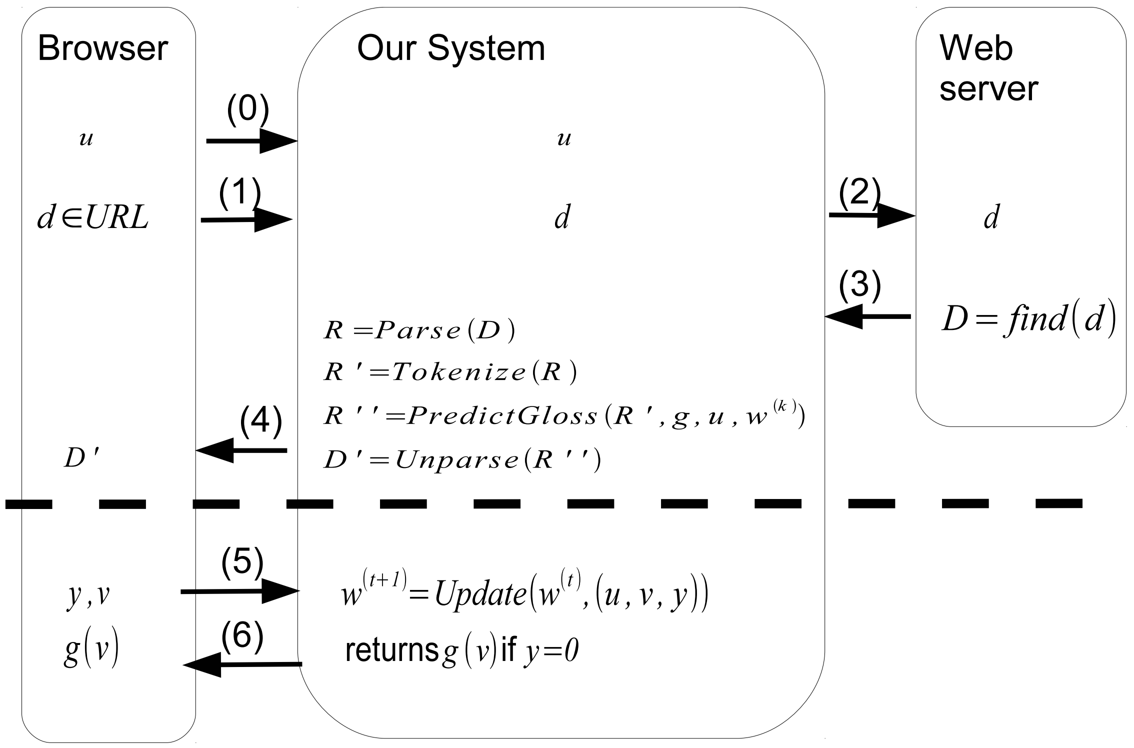


Fig. 6.2. System operation.

fails to retrieve the document, it simply returns an error message to the browser in step (4).

(4) The system extracts words from document D and returns the document with the words predicted to be unfamiliar to user u glossed with their meanings.

Web documents are usually have tags like "<html>" and "<body>". Thus, they can be represented by a tree, as shown in Figure 6.3 (a). Let Dom_D be the set of all Web pages and Dom_T be the set of trees. We define $Parse(D)$ as a function that takes a Web page $D \in Dom_D$ and returns tree $R' \in Dom_T$, that corresponds to D. Conversely, we define $Unparse(R'')$ as a function that takes tree $R'' \in Dom_T$ and returns the Web page that corresponds to R''.

Step (4) in Figure 6.2 involves a tokenization function "Tokenize" and a prediction and glossing function "PredictGloss". "Tokenize" takes tree $R \in Dom_T$ and returns a tree whose sentences are tokenized into words. An example is shown in Figure 6.3: "Tokenize" takes Figure 6.3(a) and returns the tree in Figure 6.3(b) except for the bold parts. "Tokenize" also tags every token by "<span>" and embeds a JavaScript script that enables the user to consult a dictionary simply by clicking the word.

"PredictGloss" takes tokenized tree $R' \in Dom_T$, glosser function g, user identifier u, and the weight of classifier $\boldsymbol{w}^{(t)}$ and g takes token v and returns its gloss (meaning) $g(v)$. For all the

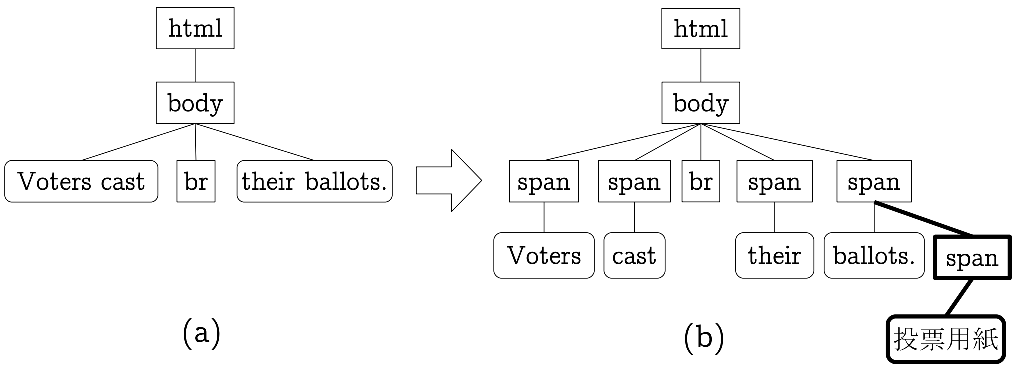


Fig. 6.3. (a) Example of a tree, (b) Tree passed back to the browser. The bold part is an example of a gloss, showing the meaning of ballots in Japanese.

leaves, i.e., tokens, in tree R', "PredictGloss" first predicts the words unfamiliar to user u and then glosses only unfamiliar ones with g. The prediction is determined by the sign of the function $h(u, v, \boldsymbol{w}^{(t)})$.

While previous systems do not require any communication with the browser after (4), our system communicates with the browser in (5) and (6) to collect the "click log" (y, v) (defined below) to train $\boldsymbol{w}^{(t)}$. This communication is made possible by the use of "AJAX", asynchronous JavaScript, and XML (extensible markup language). We used "jQuery"[*2] for the library for AJAX.

(5) When token v in a different Web document D' is clicked, the embedded JavaScript script in D' sends the pair of y and v to the system.

(6) If $y = 0$, i.e., the user does not know word v, so its gloss is sent back to user u.

Here, y codes whether or not user u knows word v. It is defined as follows: $y = 0$ is sent to the system if word v was predicted to be unfamiliar to user u prior to the click and is clicked for the first time after this prediction. $y = 0$ means that the system regards user u as not knowing word v because user u is now requesting the gloss for word v even though the system first regarded user u as knowing word v in (4). If word v is predicted to be known to user u and is clicked for the first time, the exact opposite happens: $y = 1$ is sent to the system, which means that the system regards user u as knowing word v because user u is now hiding word v even though the system first regarded user u as not knowing word v in (4). The data (u, v, y) is used to update the value of $\boldsymbol{w}^{(t)}$ for training, as shown in Figure 6.2. The function $Update(\boldsymbol{w}^{(t)}, u, v, y)$ is explained later.

[*2] http://jquery.com/

6.2.2 Implementation Details

First, note that the system should be multi-threaded and every thread processes the requests from one of the users. Simply speaking, our system involves two kinds of data: the weight vector $\boldsymbol{w}^{(t)}$ for classification, and the click logs accumulated so far to train the weight vector. These data have very different natures: the weight vector should almost always be stored in memory and accessible from every thread because it is used every classification. Moreover, it is frequently updated by the users' usage. By contrast, the click logs are, strictly speaking, not updated but rather added. Simply recording new click data suffices. Moreover, they do not need to be always loaded in memory as they are used only when the system retrains the weight vector, say during the night, for example.

Exploiting this different nature of the data, we store the weight vector in "memcached" and the clicks logs in a relational database management system (RDBMS). When the user clicks a word, the thread processing that user dispatches two requests: one to update the weight vector stored in "memcached" and one to add a new data record to the RDBMS. This "memcached" is the usual solution for this kind of environment; it is, simply speaking, a hash table shared by all the threads processing a user's request. Being a hash table—one update consists of the pair of a key (the id of a feature) and a value (the value of the feature)—means that "memcached" involves only that key and does not affect any other keys. This means that, for example, the English ability features of two different users can be simultaneously updated because they have different feature ids.

6.3 Model

The system uses the shared diffculty model introduced in §**??**. Thus, $h(u, v, \boldsymbol{w}^{(t)})$ is defined as (**??**) -0.5. Throughout this chapter, we abuse $\boldsymbol{w}$ to include user ability parameter as well. An appropriate choice of $\phi(v)$ improves accuracy. Remember that our system needs to estimate user language ability and word difficulty. Among these two, not much information about language ability is expected to be available ahead of time because many users are anonymous by the nature of the Web community. In contrast, much is known about word difficulty. While many measures of word difficulty have been proposed, most of them are based on word frequency in a large corpus because word frequency has been demonstrated to be good measure of word difficulty for ESL (English as a second language) users (Tamayo, 1987). Thus, we used two kinds of features for word difficulty: "Google 1-gram" and "SVL". Google 1-gram is created from the raw word frequency taken from Brants and Franz (2006). Taken from approximately a trillion Web pages, Brants and Franz (2006) is one of the largest word frequency lists known. Specifically, we used the value of normalized $-\log(\frac{\mathrm{freq}_v}{\sum_{v \in V} \mathrm{freq}_v})$, where freq_v is the word frequency of word v in Brants and Franz (2006). Although also based on word frequency, SVL SPACE ALC Inc. (1998) is a word difficulty measure manually checked by English native speakers for Japanese ESL users. SVL12000 covers 12,000 words and has 12 levels.

A Rasch model is not simply applicable to this system because the input data are too sparse. However, by using the notation used for logistic regression, we can tackle this problem by just adding more features to $\phi(v)$. In particular, we can approximate average word difficulty by using word frequencies; there are many report that word frequency can be used as a measure of word difficulty (Tamayo, 1987). We used word frequencies from the Google corpus Brants and Franz (2006), whose n-gram counts are claimed to have been "generated from approximately 1 trillion word tokens of text from publicly accessible Web pages".

For parameter estimation, we utilized the stochastic gradient algorithm (SGD) explained in §4.5.1 for the running system. An online algorithm suits for a Web application, because users of a Web application do not feed the whole data at a time. $Update(\boldsymbol{w}^{(t)}, u, v, y)$ in Figure 6.2 is defined by (4.42).

6.4 Experiments

This section presents the results of our experiments and discusses them.

6.4.1 Evaluation Data

The Self-report scale was used for developing the evaluation data because creating and answering multiple-choice items for 12,000 words has a much higher cost burden than the Self-report scale. We developed a database of the vocabulary knowledge of 16 human subjects for 12,000 words for training and evaluating our system. This database is a matrix of the numbers representing the degree of vocabulary knowledge, where the rows correspond to the human subjects and the columns correspond to the words. The vocabulary knowledge was obtained by assessing the *language ability of humans* introduced in Read (2000).

The database was developed by firstly determining the set of words for assessing the language ability of humans. The word set was selected from the Standard Vocabulary List 12000 (SVL12000) (SPACE ALC Inc., 1998). SVL12000 lists the most fundamental 12,000 words that an English user should learn and they have been checked by native English speakers. Then, the questions for assessing the subject's language ability were developed. Each human subject an-

Table. 6.1. Our scale for evaluation.

1	never seen the word before
2.1	probably seen the word before
2.2	absolutely seen the word before but don't know its meaning / tried to learn the word before but forgot its meaning
3	probably know the word's meaning / able to guess the word's meaning
4	absolutely know the word's meaning

swered the question for every word in SVL12000.

We created the scale shown in Table 6.1. It is based on both Dale's scale and Paribakht & Wesche's scale with two major modifications. The first modification is that we omitted rank V because that exists for testing writing ability rather than testing reading ability, which our system tries to support. The second modification is that knowledge of synonyms and translations is omitted in our ranks. The reason for this is to prevent bias in the results. As it takes time for the subjects to give a synonym or translation, they may hesitate to choose an item that requires one, especially if the test covers many words. Another modification to Dale's scale is that we divided this Rank 2 into two ranks: 2.1 and 2.2. This modification was introduced because we wanted to discriminate the case where the subject has tried to learn the word before from the case where the subject simply has seen the word before. Dale's scale does not discriminate these two because it was originally for testing L1 language ability, and words are usually learned incidentally in L1 acquisition.

16 students mostly from graduate schools of the University of Tokyo participated in the test as human subjects. Each student answered 12,000 questions corresponding to the 12,000 words using our scale (Table 6.1). Note that the ranks in Table 6.1 are numbered to correspond to those of Dale's scale (Table 3.3) and Paribakht & Wesche's scale (Table 3.4). The subjects saw a simple ordinal numbering $1, 2, 3, 4, 5$, so that numbering did not affect their results.

The scale in Table 6.1 needs to be binarized to evaluate our system because our system eventually classifies words into "known" and "unknown" to a user. This scale binarization was performed as follows: Only rank 4 in Table 6.1 was regarded as $y = 1$, i.e., the case where user u knows word v. All the other ranks were regarded as $y = 0$, i.e., the case where user u does not know word v.

The aim of this binarization is as follows: as the goal of our system is to support reading by automatically showing glosses of words, in our system's evaluation, the criteria when binarizing these scales should be whether or not the user needs the glosses of the words. Those words in ranks 1, 2.1, and 2.2 clearly need to be glossed when they appear in texts because the subject does not know them, so they surely hamper reading. The words in rank 3 also need to be glossed because, although the user may be able to "guess" their meanings, these guesses could be wrong and, without glosses being shown when the words appear in a text, the user cannot be informed of his/her wrong guesses and might not fully comprehend the text containing the words or might misinterpret the text, either of which would hamper reading.

6.4.2 Evaluation by Accuracy

The database was divided into a training set, development set, and test set. 600 words were randomly chosen for the training set, 1400 words were randomly chosen for the development set, and 9999 words were randomly chosen for the test set. Accuracy was defined as the percentage of the words that our system correctly answered among the 9999 words in the test set.

The evaluation setting simulated the case where a new user starts using our system with a specified log. Every user among the 16 subjects was assumed to start using the system as a new user and to click at most 600 words. This number 600 was chosen so that everyone could click them within about 5 minutes.

The system was trained with data from **smart.fm** Cerego Japan Inc. (2009) instead of our system's log because no log had been accumulated at the start. **smart.fm** is an implementation of a computer-assisted vocabulary acquisition (CAVOCA) system in the field of computer-assisted language learning (CALL). The purpose of a CAVOCA system is to support its users, typically ESL users, so that they can learn as many English words as possible in a fixed period. A CAVOCA system works as follows: First, the CAVOCA system gives the user a word list and the user selects words in the list to learn. If the user finds words that he/she already knows, he/she can check them and skip them. For our system, we used the checked words as training data because they are regarded as known words and the unchecked words in the word list are regarded as unknown words. Note that words absent from the word list were not used. Once the set of words to learn has been determined, the CAVOCA system automatically generates a set of questions about the words and presents these questions to the user. The CAVOCA system continues generating questions and the user continues answering them until he/she is able to answer the whole set of questions correctly. The question types include multiple-choice questions and spelling questions. While we obtained the records for 10,526 **smart.fm** users, we eventually chose 675 users from amongst these and

Table. 6.2. Values of accuracy (%) for IRT and LR.

	$N = 10$	30	100	300	600
IRT	68.33	73.69	74.65	74.65	74.60
LR	**73.25**	**77.89**	**79.09**	**80.03**	**80.01**

used their records because many users simply tried **smart.fm** and did not use it seriously and repeatedly. These 675 users were chosen according to the following criteria: selection of at least 100 words for learning and 15% or more skipped words out of the total number of words in the word lists.

Our system was trained with data created from **smart.fm** (N_0) and at most 600 new user words (N_1) as training data. Here, $N = N_0 + N_1$, where N denotes the same N as in Section 5.1. The hyperparameters were tuned for the development set, and the accuracy of our system was measured on the test set. The accuracy was defined as the average percentage for the case where the classifier predicted correctly over the test sets of the 16 subjects.

The Rasch model is a kind of IRT that can be defined as a logistic regression by providing user-IDs and word-IDs as features in logistic regression. Our model is an extension of the Rasch model with the addition of word difficulty features in logistic regression. This section describes how we evaluated the effectiveness of the word difficulty features. Fan et al. (2008) was used for both IRT and LR with L2-regularization. The regularization parameter was selected from $1.0, 0.5, 2.0$ and the value that gave the highest accuracy in the development set was used in the test. Note that since IRT is a logistic regression that does not use word difficulty features, evaluating the case where no word difficulty features are used corresponds to evaluating the effectiveness of IRT.

The results are shown in Table 6.2 and Figure 6.4. Both show that "LR", the case that includes word difficulty features, exhibited considerably higher accuracy than "IRT". This means that word difficulty features are effective because the difference between "LR" and "IRT" lies only in the word difficulty features. Thus, the use of word difficulty features is confirmed to be reasonable.

Note the difference of the curve "Rasch" in Figure **??** and the curve "IRT" in Figure 6.4: the former is more flat than the latter although both curves are produced by the Rasch model. The reason of this difference is due to the experiment setting: in "Rasch" in Figure **??** a completely out-of-sample setting is used. Thus, the Rasch model has totally no information on the new words in the test sets, resulting in the flat curve. The experiment setting used in the "IRT" curve in Figure 6.4 is, however, similar to out-of-sample setting, but not completely. That is, the Rasch model can obtain information of the new words if some other users in the "smart.fm" dataset,

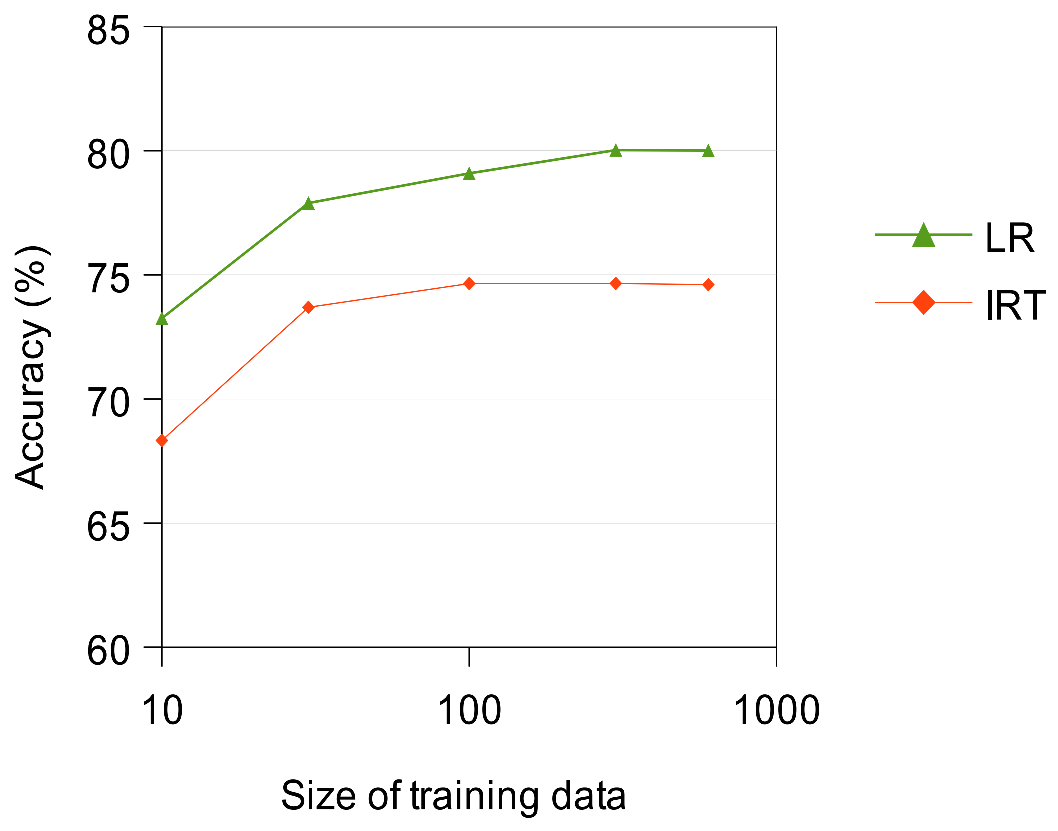


Fig. 6.4. Comparison of logistic regression and IRT.

which is included in the training data. Thus, in the "IRT" curve in Figure 6.4, the Rasch model can manage to estimate the difficulty of the new words because some of them are in the "smart.fm" dataset, a part of the training data.

Another observation from Figure 6.4 is that the accuracy of both settings seemed to saturate in the range where the size of training data was over 300. "LR" achieved about 5% higher accuracy than "IRT" as a result of using word difficulty features.

6.4.3 Comparing Logistic Regression with Other Classifiers

There are many algorithms for binary classification besides logistic regression. Our system can be trained with them by using the same features. This section compares logistic regression with other binary classification algorithms in terms of accuracy. The following algorithms were compared with logistic regression.

1. SVM (Linear)

Table. 6.3. Prediction accuracy (%) for various algorithms.

	$N_1 = 10$	30	100	300	600
SVM (Linear)	74.78	**78.08**	78.88	79.20	79.27
SVM (RBF)	67.61	77.27	**79.16**	79.55	79.91
SGD	**76.95**	77.94	78.52	78.46	78.39
LR	73.25	77.89	79.09	**80.03**	**80.01**

2. SVM (RBF)

"SVM (Linear)" is a support vector machine with a linear kernel. This algorithm was chosen for the comparison because a support vector machine is a state-of-the-art algorithm for binary classification. Unlike a support vector machine with a Gaussian kernel, a support vector machine with a linear kernel cannot fully classify nonlinear data. However, this does not cause much problem with high-dimensional data because there are enough dimensions to classify the given data linearly. Explanations of support vector machines are found in Cristianini and Shawe-Taylor (2000). Again, Fan et al. (2008) was used for the implementation. The performance of a support vector machine is known to be affected by the value of regularization parameter C. In this evaluation, C was chosen from $1.0, 2.0, 0.5$. The value was tuned for the development set.

"SVM (RBF)" is a support vector machine with a Gaussian kernel, so it is another state-of-the-art algorithm for binary classification. Fan et al. (2008) was used for the implementation. Unlike all the other algorithms, "SVM (RBF)" takes hours to optimize. The value of regularization parameter C was set to 1.0.

"Stochastic Gradient Descent" (SGD) is the SGD described so far. Note that no regularization is performed in SGD.

Finally, "LR" denotes the results for logistic regression. Again, LR was optimized with L2-regularization using Fan et al. (2008). The regularization parameter was selected from $1.0, 0.5, 2.0$, and the value that gave the highest accuracy in the development set was used for the test.

Accuracy values for logistic regression and other algorithms are listed in Table 6.3. Logistic regression achieved the highest accuracy when the number of the training data was 300 and 600. It also achieved nearly the highest accuracy for $30, 100$. This means that logistic regression is sufficiently accurate compared with the other algorithms. Thus, the use of logistic regression is reasonable for this application.

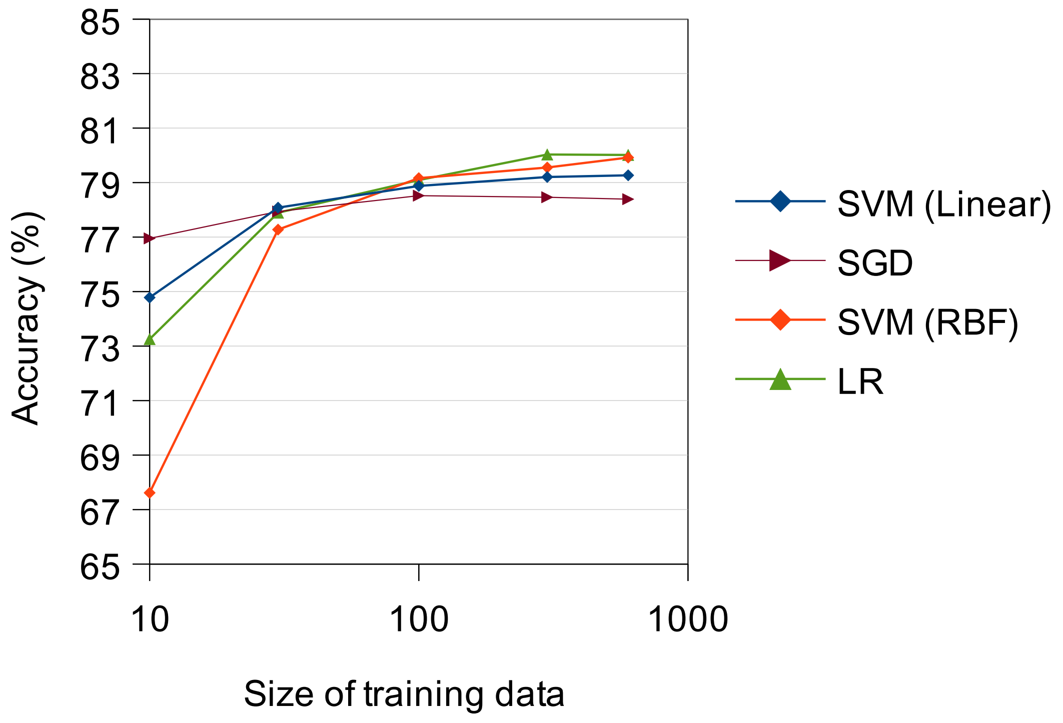


Fig. 6.5. Comparison of logistic regression and other algorithms.

When the numbers of the training data were 10 and 30, SGD outperformed LR and achieved the highest and the second highest accuracy respectively. This result is encouraging because it suggests that SGD is quick to adapt to a user. SVM (RBF) had the lowest accuracy when the size of training data was set to 10. We attributed this to the amount of training data being too small and to parameter C of the SVM (RBF) being fixed to 1.

As shown in Figure 6.5, all the algorithms saturated in the range where the size of training data exceeded 300 and even the highest of accuracy was only about 80%. This shows that about 80% is the accuracy limit owing to the nature of the given data.

6.4.4 Simulation of Reading

This section evaluates our system by simulating users' document-reading processes using the data mentioned in 6.4.1. We simulated the case where each user reads all the documents in the "Brown corpus", which consists of 500 documents taken from various sources for balance. The average number of words in a Brown Corpus document is 2029.

Leave-one-out evaluation was performed. First, we selected one user for the development to tune the parameters of the classifiers. The data from this user were used for development purposes only, and never used for training or testing. In each fold, a test user was selected, and all the data of the other users were used as training data of "training 1", i.e., to learn the word difficulty, as explained later.

1 The next document to read was given randomly.
2 Given the document to read and a test user, each classifier classified and glossed all the words in the document into known or unfamiliar.
3 Using the vocabulary knowledge collected in 6.4.1, we simulated the test user reading the documents and correcting all the words to reflect his or her knowledge of those words.
4 Each classifier was trained from these corrections. Go back to 1.

The evaluation of our system involved both its ability to estimate difficulty of words and its ability to estimate the users' language abilities. As the focus of this evaluation was the latter, we wanted to make the training data separate. Therefore, the training data were divided into two parts: "training 1" and "training 2". Here, "training 1" corresponds to the logs accumulated so far and was the part for difficulty of words.

The following three methods were compared. The C hyperparameter for the logistic regression was chosen from $100.0, 10.0, 1.0, 0.1, 0.01, 0.001$ and the value that achieved the highest accuracy (in number of types) using the data from the development user was selected. For hyperparameters of SGD, the setting $\lambda = 1.0, t_0 = 0$ was used.

Unfamiliar *The baseline*. This simulates the case where no prediction is available; i.e., the user clicks all unknown words.

LR simulates the case where in both "training 1" and "training 2", the costly batch learning of a logistic regression, is used.

LR+SGD simulates the case where the combined method is used. That is, in "training 1", a costly batch learning of a logistic regression is used. Then, the resulting weight vector (i.e., parameter vector) is used as an initial weight vector of SGD, an efficient online learning of logistic regression to tune only the weight for the test user's language ability parameter. All the other parameters including the parameters for difficulty of words are fixed during "training 2".

First, we evaluated our system in terms of the number of clicks required. Table 6.4 lists the average percentage of words to be clicked, i.e., the total number of clicks divided by the total number of words in the 500 documents. As the denominator is a constant, these percentage values

Table. 6.4. Average percentage of words to be clicked in a document in the corpus consisting of 500 documents.

User	Unfamiliar (%)	LR	LR+SGD
user0	64.74	**25.51** ***	29.89 ***
user1	21.21	**12.36** ***	12.80 ***
user2	12.55	**10.25** ***	11.58 ***
user3	11.89	**6.93** ***	7.62 ***
user4	9.82	**7.09** ***	7.51 ***
user5	7.33	**5.65** ***	6.94 *
user6	5.33	**4.03** ***	4.98 *
user7	5.32	**4.55** ***	5.15
user8	**5.10**	5.52 ***	6.68 ***
user9	4.56	**4.53**	5.77 ***
user10	4.34	**4.02** ***	5.07 ***
user11	**1.80**	2.42 ***	3.26 ***
user12	**1.51**	2.08 ***	2.79 ***
user13	**1.08**	2.05 ***	2.56 ***
user14	**0.39**	2.06 ***	2.84 ***

are proportional to the required number of clicks. Note that, here, the number of words means the number of occurrences of words, not the number of types of words as used in the previous section. The users were sorted by the percentage of their unfamiliar words in descending order. The asterisks are T-test trial results for each method versus "Unfamiliar", the baseline. "*" denotes that the p-value is lower than 0.05, "**" denotes that the p-value is lower than 0.01, and "***" denotes that the p-value is lower than 0.001.

Investigating Table 6.4, we can find that, from user0 to user7, "LR" significantly reduced the number of clicks and from user0 to user6, "LR+SGD" significantly reduced the number of clicks. Intuitively, the smaller the number of the user's unfamiliar words becomes, the more difficult for the system to make significant reductions in the number of clicks. However, this case is not problematic for the user because, if the user has a large enough vocabulary to read the documents without any support, the user does not need reading support in the first place.

Readers do not always need to be familiar with all the words in a document to read that document. Nation (2006) surveyed and investigated the percentage of words necessary to read docu-

Table. 6.5. Average percentages of remaining unfamiliar words in the 500 documents.

User ID	Unfamiliar (%)	LR	LR+SGD
user0	64.74	**14.52** ***	14.66 ***
user1	21.21	7.73 ***	**6.07** ***
user2	12.55	6.28 ***	**4.82** ***
user3	11.89	4.59 ***	**3.26** ***
user4	9.82	4.26 ***	**3.25** ***
user5	7.33	3.34 ***	**2.56** ***
user6	5.33	2.30 ***	**1.71** ***
user7	5.32	2.54 ***	**1.73** ***
user8	5.10	2.91 ***	**2.38** ***
user9	4.56	2.42 ***	**1.93** ***
user10	4.34	2.13 ***	**1.66** ***
user11	1.80	0.76 ***	**0.57** ***
user12	1.51	0.54 ***	**0.43** ***
user13	1.08	0.38 ***	**0.32** ***
user14	0.39	0.17 ***	**0.13** ***

ments and reported that one must know from 95% to 98% of the words in a running text to read documents sufficiently without assistance.

We also evaluated our system in terms of the percentage of "remaining unfamiliar words", as shown in Table 6.5. As the words predicted to be unknown to a user were glossed in advance, the users could know the meaning of those glossed words when using our system. Thus, the unfamiliar words remaining were the words predicted to be known to a user, though the user actually did not know them. We call these words "remaining unfamiliar words". The meaning of asterisks is the same as in the previous table.

From Table 6.5, we can see that the numbers of remaining unfamiliar words were significantly decreased for all users. In particular, for user2–to user8, the system significantly made unreadable documents nearly readable as the average remaining unfamiliar became lower than 5% while originally being above 5%, according to Nation (2006). Moreover, interestingly, the proposed "LR+SGD" achieved better results than the costly "LR" method. This is probably due to the fact that in "LR+SGD", SGD updates only the language ability parameters while "LR" tries to adjust all the parameters and causes overtraining.

Finally, we compared the time required to train the logistic regression in “LR” and SGD in “SGD”. While the total time to finish the training for 500 documents was 0.43 (s) in SGD on average, LR took 1975.53 (s) and the t-test was of course significant as the p-value was under 10^{-22}. We can easily see that “LR+SGD” was the most efficient.

6.5 Chapter summary

We described a new glossing system that can predict words unknown to the user by utilizing logs that contain valuable information about a user’s vocabulary. Although existing glossing systems are helpful for English users because they provide the meaning of a word by displaying it in a pop-up window when the user encounters an unknown word and clicks on or mouses over it, they waste this log information.

We investigated models for our system’s prediction. The use of logistic regression was found to be appropriate because the Rasch model, a simple form of item response theory (IRT) widely used to assess human language ability, is also a logistic regression that uses a specific feature vector. We extended IRT by introducing word difficulty features and achieved accuracy higher than that of straightforward IRT.

We also proposed a method of training parameters suitable for our system. IRT and the extended IRT can estimate both the difficulty of words and the users’ language abilities simultaneously from the accumulated logs as one parameter vector. Note that the estimated difficulty of the words is not simple usage of existing word difficulty measures, but is adjusted using the combination of existing word difficulty measures to fit the training data from the system users. For example, if an existing word difficulty measure is not correlated with the system users, the weight word difficulty measure approaches zero and is nearly omitted.

Our system requires two kinds of estimation: one is estimation of the difficulty of the words so that it suits the users of our system; the other is estimation of user language levels including those of new users. The former is more stable because the difficulty of a word does not change suddenly, while the latter is more dynamic because user language levels differ from user to user and our system needs to become accustomed to a new user quickly.

To meet these requirements, the proposed estimation method combines both batch learning and online learning by sharing the parameter vector. The costly batch learning is used to learn the difficulty of words while our system is idle, and the estimated parameter is copied to the stochastic gradient descent (SGD) user. SGD updates only the users’ language ability parameters quickly in an online manner and is thus appropriate for our system where the logs are accumulated in an online manner.

We evaluated our system in terms of the number of clicks and the percentage of unfamiliar words remaining even after the prediction and glossing. To evaluate our system, we collected knowledge about 12,000 words from 16 participants and used these data to simulate users' reading of the Brown corpus.

The simulation results show that our system can significantly reduce the number of clicks for users who know up to 94.7% of the running words in a document (a.k.a., coverage). As Nation (2006) shows that a reader should have 95% or higher coverage of a document to read it sufficiently, this shows that our system can reduce the number of clicks for most readers with insufficient word coverage to read documents. The simulation results also show our system can significantly reduce the number of remaining unfamiliar words after the prediction and glossing for all users.

Finally, we compared the time required to train the costly batch model and our combined method. As the latter took 0.43 s for the training of 500 documents on average while the former took 1975.53 s, the effectiveness of the combined method is shown.

To conclude, the combined use of batch learning for mainly learning word difficulty and online learning for mainly learning the users' language ability is efficient for increasing the training speed, decreasing the required number of clicks, and decreasing the number of unfamiliar words.

Chapter 7

Conclusion

The proposed unified framework applies language testing technology to practical systems for supporting second language users. Language testing methods provide a methodology for measuring a user's second language knowledge by analyzing the user's responses to stimuli. However, language testing methods cannot *support* second language users. Support systems are designed to do this; however, because they are not equipped with a methodology for measuring a user's second language knowledge, they cannot be used to measure and use it. Therefore, the frameworks for these two fields have been separate. Unlike the case with a native language, the skills of second language users vary greatly, so a support system that includes second language knowledge measurement is very promising, particularly in terms of personalizing the support. We categorized the tasks to which the proposed framework is naturally applicable and identified the properties that models under the proposed framework should have. As vocabulary knowledge is the basis of second language knowledge, we focused on vocabulary studies and vocabulary testing within second language knowledge.

The tasks with which support systems deal vary. To identify the tasks to which the proposed framework is applicable, we introduced the following categorization of tasks under the proposed framework: *response modeling*, *support modeling*, and *stimuli modeling*. We categorized these tasks by using the categorization in §2. We conclude that language testing technology can be naturally applied to tasks that can be categorized into response modeling from the stimuli category. We then identified the properties that language testing models should have to be used in practical application systems. The desired properties are *interpretable parameters using language testing models*, *globally optimal parameter estimation*, *out-of-sample setting*, and *noisy stimuli (word) detection*.

Among language testing methods, detailed vocabulary assessment methods were surveyed in §3. We especially focused on the *Vocabulary Size Test* (Nation and Beglar, 2007) and *Yes/No* test

(Meara, 1992). We found that both tests sample words to use in tests on the basis of word frequency. The *Vocabulary Size Test* tests vocabulary knowledge by using a multiple-option format, and the Yes/No test uses a self-report format. To prevent overestimation of vocabulary, pseudowords are used in the Yes/No test, and the formulae used to estimate vocabulary size from this test format have become sophisticated. However, that many studies argued that there are shortcomings in the use of pseudowords and that many reports argue that the estimated vocabulary sizes do not change greatly.

We surveyed the *item response theory* models, the core of sophisticated language testing technology, in 4. The two- and three-parameter IRT models are not convex while they can detect noisy words. Only the Rasch model and the shared difficulty model are convex but they cannot detect noisy words. Thus, IRT models do not have all the desired properties.

In §5, we presented our proposed convex model and showed that its accuracy is better than that of previous models. We also proposed a method for calculating user-specificity, which can be used as a measure of word noisiness when predicting vocabulary. The proposed model is convex and can successfully detect noisy words. Thus, it has all the desired properties. Moreover, we showed that the proposed model can outperform the two-parameter IRT model, the previous model that supports noisy word detection, by over 10% by using the practical size of users' responses as training data in the out-of-sample setting. We introduced the generalization of the proposed model by using graph Laplacian matrices as well.

Finally, in §6, we demonstrated our framework for reading support by using language testing technology on vocabulary. Through a simulated use case, we demonstrated that the novel system built under the proposed framework can reduce the number of clicks and increase the number of readable documents if the user's ability is sufficiently low. For this purpose, we created the first dataset of second language users' knowledge. This dataset covers a large body, 11,999 words, of English vocabulary.

The contributions of this thesis can be summarized as follows:

1. This thesis is the first attempt at creating a framework that can apply language testing models to practical systems for supporting second language users.
2. To this end, we proposed an extension of the Rasch model, one of the previous language testing models, that can work robustly in practical systems by satisfying all the desired properties.
3. We demonstrated the use of our framework for reading support that can detect words unfamiliar to users and automatically translate them.

4. For our demonstration, we created the first dataset of second language users' knowledge that covers a large body of English vocabulary.

Future work under the proposed framework includes semi-supervised prediction and active learning to reduce the number of training labels.

Achievements

Refereed achievements related to this thesis:

Journals

Yo Ehara, Nobuyuki Shimizu, Takashi Ninomiya, Hiroshi Nakagawa. Personalized Reading Support for Second-Language Web Documents. ACM Transactions on Intelligent Systems and Technology, 4(2), 2013. (Journal Version of IUI Paper, Chapter 6)

Conferences

Yo Ehara, Issei Sato, Hidekazu Oiwa, Hiroshi Nakagawa. Mining words in the minds of second language learners: learner-specific word difficulty. In the Proceedings of the 24th International Conference on Computational Linguistics (COLING-2012). pp. 799–814 (Long paper). Mumbai, India, December 2012. (Chapter 5)

Yo Ehara, Nobuyuki Shimizu, Takashi Ninomiya, Hiroshi Nakagawa. Personalized Reading Support for Second-Language Web Documents by Collective Intelligence. In the Proceedings of the 2010 International Conference on Intelligent User Interfaces (IUI 2010). pp. 51–60. Hong Kong, China, February 2010. (acceptance rate 22%)

Refereed other achievements:

Journals

江原遥, 田中久美子. TypeAny:言語判別を用いた多言語入力システム. 自然言語処理. 15(5):151–168, October 2008.

Conferences

Yo Ehara and Kumiko Tanaka-Ishii. Multilingual Text Entry using Automatic Language Detection. In the Proceedings of the 3rd international Joint Conference on Natural Language Processing (IJCNLP 2008). pp. 441–448. Hyderabad, India, January 2008. (acceptance rate 28%)

Acknowledgments

I deeply appreciate my supervisor Prof. Hiroshi Nakagawa. He has been tolerant and patient in advising and encouraging me in spite of my sometimes disappointing results and behavior throughout these long years. Without his help, I would not have been able to write this thesis. I also appreciate my thesis committee members: Prof. Kenji Yamanishi, Prof. Hisashi Kashima, Prof. Akiko Aizawa, and Prof. Hiroya Takamura. Their corrections and fundamental questions were quite helpful for improving this thesis.

I would like to thank Prof. Takashi Ninomiya for supervising my master's thesis. This master's thesis was the basis of §6. He also advised me to create an exhaustive dataset, which this thesis largely depends on. I would like to thank Prof. Issei Sato for giving me insightful comments on the later part of my research. I would like to thank all my laboratory members, especially Mr. Hidekazu Oiwa and Dr. Shin Matsushima for their insightful discussions. Mr. Hidekazu Oiwa is the third author of my COLING 2012 paper, on which §5 is largely based, and he thankfully shared his precious time with me for my paper at the time of submission, although we were busy attending the Machine Learning Summer School 2012 held in Kyoto. I would like to thank my family for patiently and kindly supporting me in completing this thesis.

I would like thank Prof. Satoshi Sekine for mentoring me while I was an intern at Rakuten Institute of Technology - New York (RIT-NY). I greatly enjoyed life in New York. His insightful comments greatly helped to change my machine-learning-centered view of NLP to more data-driven view by "looking at the data".

Finally, I also would like to show my gratitude to the medical doctors for having saved my life when I was an infant. I was born with a birth defect of the large arteries of the heart called dextro-Transposition of the Great Arteries (d-TGA) Type II and fully recovered by undergoing Jatene surgery, which was at the experimental stage when I had the surgery at the age of one. I am one of the eldest Japanese who underwent this surgery. From 1982 to June 1985, twenty two infants, boys, and girls with d-TGA Type II underwent Jatene surgery at the Tokyo Women's Medical University. Six died and sixteen survived (Table 2, 石原 和明 (1986)). I am one of the sixteen survivors. At that time, many of the infants born with this defect were able to safely enjoy

their teenage years, or maybe twenties, if they chose to undergo palliative operation. Our parents, however, chose to try the Jatene surgery for the sake of their children's full recovery.

Trying is the base of science: it requires braveness. This anecdote has been motivating me to pursue my Ph. D. course. Now fully recovered, I wrote this anecdote to remember our parents' braveness and hoping that future infants born with this defect can overcome it and lead fulfilling lives.

Bibliography

T. Abekawa, N. Kikuko, M. Okumura, Y. Yagi, N. Totsugi, S. Sugimoto, and L. Fu. Development of japanese reading system of multi-lingual environment ” asunaro” . In *Proceedings of the 3rd International Conference on Computer Assisted Systems for Teaching and Learning/Japanese (CASTEL-J)*, pages 71–74, 2002.

S. Amano and T. Kondo. Estimation of mental lexicon size with word familiarity database. In *Proceedings of the 5th International Conference on Spoken Language Processing (ICSLP)*, 1998.

R. C. Anderson and P. Freebody. Reading comprehension and the assessment and acquisition of word knowledge. In B. Huston, editor, *Advances in reading/language research*, pages 132–255. JAI Press, 1983.

F. B. Baker and S.-H. Kim. *Item Response Theory: Parameter Estimation Techniques*. Marcel Dekker, New York, second edition, 2004.

T. Baldwin, V. Kordoni, and A. Villavicencio. Prepositions in applications: A survey and introduction to the special issue. *Computational Linguistics*, 35(2):119–149, 2009.

Z. Bao, B. Kimelfeld, and Y. Li. A graph approach to spelling correction in domain-centric search. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 905–914, Portland, Oregon, USA, June 2011.

D. Beglar. A rasch-based validation of the vocabulary size test. *Language Testing*, 27(1):101–118, 2010.

C. M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

R. D. Bock and M. Aitkin. Marginal maximum likelihood estimation of item parameters: Application of an em algorithm. *Psychometrika*, 46:443–459, 1981.

R. D. Bock and M. Liberman. Fitting a response model for n dichotomously scoreditems. *Psychometrika*, 35:179–197, 1970.

S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

T. Brants and A. Franz. Web 1T 5-gram Version 1, 2006. LDC2006T13.

Cerego Japan Inc. smart.fm, 2009. System available at `http://smart.fm/`.

K. Collins-Thompson and J. Callan. Predicting reading difficulty with statistical language models. *Journal of the American Society for Information Science and Technology*, 56(13):1448–1462, Nov. 2005.

Coolest.com Inc. popjisyo.com, 2002. `http://www.popjisyo.com/`.

N. Cristianini and J. Shawe-Taylor. *An introduction to support Vector Machines: and other kernel-based learning methods*. Cambridge Univ Pr, 2000.

D. Dahlmeier and H. T. Ng. Grammatical error correction with alternating structure optimization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 915–923, Stroudsburg, PA, USA, 2011a. Association for Computational Linguistics.

D. Dahlmeier and H. T. Ng. Correcting semantic collocation errors with l1-induced paraphrases. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 107–117, Edinburgh, Scotland, UK., July 2011b.

E. Dale and J. S. Chall. A formula for predicting readability. *Educational Research Bulletin*, 27 (1):11–20, 28, 1948.

M. Davies. N-grams data from the corpus of contemporary american english (coca), 2011. Downloaded from `http://www.ngrams.info` on June 23, 2012.

J. Davis. Facilitating effects of marginal glosses on foreign language reading. *Modern Language Journal*, pages 41–48, 1989.

A. M. B. de Groot and R. Keijzer. What is hard to learn is easy to forget: The roles of word concreteness, cognate status, and word frequency in foreign-language vocabulary learning and forgetting. *Language Learning*, 50(1):1–56, March 2000.

Y. Ehara, N. Shimizu, T. Ninomiya, and H. Nakagawa. Personalized reading support for second-language web documents by collective intelligence. In *Proceedings of the 15th international conference on Intelligent user interfaces (IUI 2010)*, pages 51–60, Hong Kong, China, 2010. ACM.

J. Eyckmans. *Measuring receptive vocabulary size : reliability and validity of the yes/no vocabulary test for French-speaking learners of Dutch*. PhD thesis, Radboud University Nijmegen, 2004.

R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. LIBLINEAR: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008.

G. Fischer. Logistic latent trait models with linear constraints. *Psychometrika*, 48(1):3–26, 1983.

R. Flesch. A new readability yardstick. *Journal of applied psychology*, 32(3):221, 1948.

T. François and C. Fairon. An "ai readability" formula for french as a foreign language. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 466–477, Jeju Island, Korea, July 2012.

M. Gamon. Using mostly native data to correct errors in learners' writing. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, pages 163–171, Los Angeles, California, June 2010.

N.-R. Han, M. Chodorow, and C. Leacock. Detecting errors in english article usage by non-native speakers. *Natural Language Engineering*, 12(2):115–129, June 2006. ISSN 1351-3249. doi: 10.1017/S1351324906004190. URL `http://dx.doi.org/10.1017/S1351324906004190`.

B. A. Hanson. Irt command language (icl).

Y. Hayashibe, M. Hagiwara, and S. Sekine. phloat : Integrated writing environment for esl learners. In *Proceedings of the 2nd Workshop on Advances in Text Input Methods (WTIM 2)*, pages 57–72, Mumbai, India, December 2012.

M. Heilman, K. Collins-Thompson, and M. Eskenazi. An analysis of statistical models and features for reading difficulty prediction. In *Proceedings of the 3rd Workshop on Innovative Use of NLP for Building Educational Applications (BEA)*, pages 71–79, Columbus, Ohio, June 2008.

C.-c. Huang, P.-c. Yang, K.-j. Chen, and J. S. Chang. Transahead: A computer-assisted translation and writing tool. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 352–356, Montréal, Canada, June 2012a.

C.-c. Huang, P.-c. Yang, M.-h. Chen, H.-t. Hsieh, T.-h. Kao, and J. S. Chang. Transahead: A writing assistant for cat and call. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 16–19, Avignon, France, April 2012b.

I. Huibregtse, W. Admiraal, and P. Meara. Scores on a yes-no vocabulary test: correction for guessing and response style. *Language Testing*, 19(3):227–245, 2002.

J. H. Hulstijn. Intentional and incidental second language vocabulary learning: reappraisal of elaboration, rehearsal and automaticity. In P. Robinson, editor, *Cognition and Second Language Instruction*, chapter 9, pages 258–286. Cambridge University Press, 2001.

J. H. Hulstijn, M. Hollander, and T. Greidanus. Incidental vocabulary learning by advanced foreign language students: The influence of marginal glosses, dictionary use, and reoccurrence of unknown words. *The Modern Language Journal*, 80(3):327–339, 1996.

Y. Kawamura and T. Kitamura. Reading tutor, 1997. System available at `http://language.tiu.ac.jp/`.

K. Kireyev and T. K. Landauer. Word maturity: Computational modeling of word knowledge. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 299–308, Portland, Oregon, USA, June 2011.

K. Knight and I. Chander. Automated postediting of documents. In *Proceedings of the National Conference on Artificial Intelligence*, pages 779–779. JOHN WILEY & SONS LTD, 1994.

T. Kudo and Y. Matsumoto. Japanese dependency analysis using cascaded chunking. In *Proceedings of the 6th Conference on Natural Language Learning 2002 (CoNLL, COLING 2002 Post-Conference Workshops)*, pages 63–69, 2002.

J. Lee. Automatic article restoration. In *Proceedings of Human Language Technologies: The 2004 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL): Student Research Workshop*, pages 31–36, Boston, Massachusetts, USA, May 2004.

J. Lee and S. Seneff. Correcting misuse of verb forms. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 174–182, Columbus, Ohio, June 2008.

C. Lin, R. Weng, and S. Keerthi. Trust region Newton method for logistic regression. *The Journal of Machine Learning Research*, 9:627–650, 2008.

D. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1):503–528, 1989.

X. Liu, B. Han, and M. Zhou. Correcting verb selection errors for esl with the perceptron. In *Proceedings of the 12th international conference on Computational linguistics and intelligent text processing (CICLING) - Volume Part II*, pages 411–423, Berlin, Heidelberg, 2011.

L. Lomicka. To gloss or not to gloss: An investigation of reading comprehension online. *Language Learning and Technology*, 1(2):41–50, 1998.

Q. Ma, N. Koichi, M. Murata, and H. Isahara. Selection of japanese-english equivalents by integrating high-quality corpora and huge amounts of web data. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC)*, 2008.

D. McNicol. *A Premier of Signal Detection Theory*, chapter Threshold theory, pages 136–137. Psychology Press, 2005.

P. Meara and B. Buxton. An alternative to multiple choice vocabulary tests. *Language Testing*, 4 (2):142–154, 1987.

P. M. Meara. *EFL Vocabulary Tests (First Edition).* Lognostics (Center for Applied Language Studies, University of Wales), Swansea, 1992.

P. M. Meara. *EFL Vocabulary Tests (Second Edition).* Lognostics (Center for Applied Language Studies, University of Wales), Swansea, 2010.

G. Minnen, F. Bond, and A. Copestake. Memory-based learning for article generation. In *Proceedings of the 5th Workshop on Computational Language Learning (CoNLL)*, pages 43–48, 2000.

R. J. Mislevy. Bayes modal estimation in item response models. *Psychometrika*, 51:117–195, 1986.

K. Mochida and M. Harrington. The yes/no test as a measure of receptive vocabulary knowledge. *Language Testing*, 23(1):73–98, 2006.

R. Nagata, K. Morihiro, A. Kawai, and N. Isu. A feedback-augmented method for detecting errors in the writing of learners of english. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics (COLING-ACL)*, pages 241–248, 2006.

R. Nagata, J.-i. Kakegawa, H. Sugimoto, and Y. Yabuta. Recognizing noisy romanized japanese words in learner english. In *Proceedings of the 3rd Workshop on Innovative Use of NLP for Building Educational Applications (BEA)*, pages 27–35, Columbus, Ohio, June 2008.

I. Nation and D. Beglar. A vocabulary size test. *The Language Teacher*, 31(7):9–13, 2007.

I. S. P. Nation. How large a vocabulary is needed for reading and listening? *Canadian Modern Language Review*, 63(1):59–82, 2006.

P. Nation. Vocabulary size test (monolingual, 20,000, version a). Available from `http://www.victoria.ac.nz/lals/about/staff/paul-nation`, 2007. Retrieved in Mar. 4, 2013.

P. Nation. The bnc/coca word family lists, September 2012a.

P. Nation. Vocabulary size test instructions and description. Available from `http://www.victoria.ac.nz/lals/about/staff/paul-nation`, 10 2012b. Retrieved in Mar. 4, 2013.

P. Nation and R. Waring. Vocabulary size, text coverage and word lists. *Vocabulary: Description, acquisition and pedagogy*, pages 6–19, 1997.

D. Nobori. Zurukko!, 2010. `http://zurukko.jp/`.

Nori. Firedictionary.com, 2005. `http://www.firedictionary.com/`.

A. B. Novikoff. On convergence proofs for perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata*, volume 12, pages 615–622, 1963.

B. Pang and L. Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 115–124, Ann Arbor, Michigan, June 2005.

Y. A. Park and R. Levy. Automated whole sentence grammar correction using a noisy channel model. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 934–944, Portland, Oregon, USA, June 2011.

A. Pellicer-Sànchez and N. Schmitt. Scoring yes-no vocabulary tests: Reaction time vs nonword approaches. *Language Testing*, 29(4):471–488, 2012.

popIn Inc. popin, 2008. `http://www.popin.cc/en/home.html`.

J. Read. *Assessing Vocabulary*. Cambridge University Press, 2000.

J. Read and C. A. Chapelle. A framework for second language vocabulary assessment. *Language Testing*, 1:1–32, 2001.

A. Rozovskaya and D. Roth. Algorithm selection and model adaptation for esl correction tasks. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 924–933, Stroudsburg, PA, USA, 2011. ISBN 978-1-932432-87-9.

S. E. Schwarm and M. Ostendorf. Reading level assessment using support vector machines and statistical language models. In *Annual Meeting of the ACL*, pages 523–530, 2005.

L. Si and J. Callan. A statistical model for scientific readability. In *Proceedings of the 10th International Conference on Information and Knowledge Management (CIKM)*, pages 574–576, 2001.

SPACE ALC Inc. Standard vocabulary list 12,000, 1998.

R. Stubbe. Do pseudoword false alarm rates and overestimation rates in yes/no vocabulary tests change with japanese university students' english ability levels? *Language Testing*, 29(4): 471–488, 2012.

T. D. Rudick. Rikai, 2001. `http://www.rikai.com/`.

T. Tajiri, M. Komachi, and Y. Matsumoto. Tense and aspect error correction for esl learners using global context. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL, Volume 2: Short Papers)*, pages 198–202, Jeju Island, Korea, July 2012.

J. M. Tamayo. Frequency of use as a measure of word difficulty in bilingual vocabulary test construction and translation. *Educational and Psychological Measurement*, 47(4):893–902, 1987.

K. Tanaka-Ishii, S. Tezuka, and H. Terada. Sorting texts by readability. *Computational Linguistics*, 36(2):203–227, 2010.

J. R. Tetreault and M. Chodorow. The ups and downs of preposition error detection in ESL writing. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING)*, pages 865–872, Manchester, UK, August 2008.

The BNC Consortium. The british national corpus, version 3 (bnc xml edition), 2007. Distributed by Oxford University Computing Services on behalf of the BNC Consortium. URL: `http://www.natcorp.ox.ac.uk/` (Retrieved on October 26, 2012).

H. Toyoda. *Koumoku Hannou Riron [Riron hen] - Tesuto no Suuri - (in Japanese).* Asaku, 2005.

H. Toyoda. *Koumoku Hannou Riron [Nyuumon Hen] (in Japanese, Second Edition).* Asakura Shoten, 2012.

M. West. *A General Service List of English Words.* Longman, Green & Co., London, 1953.

T. D. Wickens. *Elementary Signal Detection Theory*, chapter 8. FINITE-STATE MODELS, pages 140–142. Oxford University Press, 2001.

I. Yanguas. Multimedia glosses and their effect on l2 text comprehension and vocabulary learning. volume 13, pages 49–67, 2009.

Y. Zhang and D.-Y. Yeung. A convex formulation for learning task relationships in multi-task learning. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 733–742, 2010.

J. Zimmerman, P. K. Broder, J. J. Shaughnessy, and B. J. Underwood. A recognition test of vocabulary using single-detection measures, and some correlated of word and nonword recognition. *Intelligence*, 1:5–31, 1977.

吉見 毅彦, 小谷 克則, and 九津見 毅. テキスト言語的特徴と英語学習者の英文読解能力に基づく英文読解時間予測モデル. **教育システム情報学会誌**, 25(3):272–281, 2008. ISSN 13414135. URL `http://ci.nii.ac.jp/naid/40016433956/`.

石原 和明. 完全大血管転位症における jatene 手術の位置づけに関する臨床的検討. **東京女子医科大学雑誌**, 56(2):221–231, 1986.

Appendix A

Appendix is not public for journal publication and copyright issues.