修士論文

言語的作問技法を用いたCAPTCHAの構築

Design and Implementation of CAPTCHA using Verbal Test

筑波技術大学大学院 技術科学研究科 保健科学専攻
修士課程 2012年度入学 学籍番号 123202

山口　通智

指導教員　（主）岡本 健 （副）佐々木 信之

2014年2月20日

あらまし

CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) は、人間とロボットを判別するチューリングテストの一種である。ここでロボットとは、人工知能による自動プログラムを意味する。CAPTCHA は、ロボットにより大量のアカウントを自動的に生成するなどの不正防止を目的としており、多くのウェブサイトで利用されている。しかしながら、現在主流となっている画像型 CAPTCHA は、視覚障害者には利用できない。代替として提案された音声型 CAPTCHA は、ロボットのみならず人間にも難しく、実用的でない。

本研究の目的は、知覚に関してバリアフリーな CAPTCHA の構成である。

本研究では、バリアフリーな CAPTCHA の要求仕様として、(1) バリアフリー要件、(2) 知識非依存性要件、(3) 問題新規性要件、(4) 識別性要件、を定義し、それぞれについて解決策を提示する。(1) と (2) については、 常識に基づき文意や文脈を解釈する問題を用いる。この種の問題は、文字情報として提示できればよいので、利用者の希望する知覚で情報を伝達できる。(3) については、ウェブ上にある大量の文章を作問の種とすることで、常に新しい作問をおこなう。問題の使い回しを避けることで、ロボットによる解答集を用いた攻撃に対抗する。これは、少量の秘匿文章を用いる既存方式では達成できない。一方で、公開文章を利用するため、ロボットが検索により、解答のヒントを得る危険がある。これでは要件 (4) を満たさない。この対策として、本研究では、問題として提示する文の子音を改変する。改変された文は、形態素解析が正しくおこなわれないため、音声認識などによる文の修復が困難になる。元の文が秘匿されるので、ロボットが提案方式を検索により攻撃することは困難である。

本稿では、具体的な文意や文脈の解釈問題として、(a) ワードサラダ文と自然文の識別、(b) 機械翻訳文と自然文の識別、(c) 文章に共通する話題の識別、以上の 3 方式を提案する。さらに、これらのプログラムを実装し、(1) 被験者によるチューリングテストとしての能力、(2) 問題新規性、(3) 検索攻撃への耐性、についての評価をした。(1) に関して、視覚障害者を被験者とした実験により、最新の Google 音声型 CAPTCHA に比べて、提案方式 (a), (c) が高い識別能力を持つことを示した。(2) については、いずれの方式も 99% を超える新規問題文生成能力を持つことを示した。(3) については、現実的な検索攻撃に対して、耐性があることを示した。

Abstract

CAPTCHA (Completely Automated Public Turing test to tell Computers and Human Apart) is now the de facto standard security technology to protect on-line registration systems from malicious software. CAPTCHA systems generate several kinds of AI (Artificial Intelligence) problems which are difficult for software agents but easy for humans.

A big social problem is that most visually-impaired people cannot pass current CAPTCHAs. Most conventional CAPTCHAs employ AI problems requiring visual recognition, so they are not accessible to visually-impaired people. Audio CAPTCHAs are an alternative to visual ones, but several researchers have pointed out that state-of-the-art audio ones are too difficult for visually impaired people.

In this paper, we propose a new CAPTCHA system, which generates tests in verbal style. Our CAPTCHA system can differentiate between humans and software agents pretending to be humans, based on their different contextual cognition. It therefore works without relying on the specific perceptual abilities of the users. In our test, we utilize open documents for material of the tests. Note that there is quite a large amount of documents on the web, so we can generate brand-new tests every time. This is different from conventional studies.

One criticism is that adversaries can look for the phrases of the tests from the Internet and obtain several hints. Our system hides the sources by substituting the consonants of the phrases against such adversaries. The mechanism is similar to the phenomenon of “consonant gradation” in natural languages. The substitutes make it harder for adversaries to look for the sources because they have difficulty finding the original phrases from the erroneous ones.

We apply our idea to three kinds of verbal tests: (a) *Markov-chain Phrase Test*, which involves distinguishing between natural and machine-synthesized phrases, (b) *Machine-translated Phrase Test*, which involves distinguishing between natural and machine-translated phrases, and (c) *Topic Detection Test*, which is a choice of the common topic from several short-texts. We then implement them as CAPTCHA programs, and evaluate their performances as follows:

(1) ability to be used for a Turing test, which must be easy for humans to solve,

(2) ability to generate new tests without limitation on amount, and

(3) ability to hide the sources of the phrases which appear in the test.

Consequently, we have clarified the feasibility of our proposal as CAPTCHA for the visually-impaired.

目 次

付 図

付 表

1 序論

1.1 CAPTCHA

CAPTCHA[36] (Completely Automated Public Turing test to tell Computers and Humans Apart) は、人間と自動プログラム（ロボット）を判別する完全に自動化された公開チューリングテストである。このテストは、今では事実上の標準的なコンピュータセキュリティ技術の 1 つとして、多くのオンラインサービスに導入されている。次に、その例を挙げる。

ブログなどにおけるコメントスパムの防止: コメントスパムとは、ロボットによりなされる、コメント欄の荒らし行為やサーチエンジンのランキング向上を狙った偽のコメント投稿を指す。CAPTCHA の利用により、ロボットによる不正なコメントの投稿を抑制できる。

オンライン登録の保護: 近年では、多くのサービスがオンライン上で提供されている。代表的な例にフリーメールサービスがあるが、CAPTCHA の導入以前は「ボット」と呼ばれる不正プログラムにより、毎分数千ものアカウントが不正に取得され、スパム業者に利用されていた。CAPTCHA の利用により、ロボットによるサービスの不正利用を抑制できる。

スパム業者からの E-mail アドレスの保護: スパム業者は、ウェブをクローリングすることで E-mail アドレスを収集する。CAPTCHA のように、ロボットに解釈が困難な形式で E-mail アドレスを提示することで、クローラによる E-mail アドレスの収集を避ける事ができる。

オンライン投票の保護: ロボットによる投票が可能ならば、多重投票による不正な意見操作の恐れがあるため、投票結果の信頼性が失われる。CAPTCHA の利用により、人間によってなされた投票結果であることを保証できる。

辞書攻撃の防止: 辞書攻撃とは、単語などの文字列を辞書データとして持ち、それらやその組み合わせを入力してパスワードを破る方法である。これは、パスワードの候補となる文字列全てを入力する総当り攻撃に比べ、短時間で攻撃を試すことができる。パスワード入力前に CAPTCHA を解かせることで、ロボットによるパスワード入力を妨害し、辞書データとの照合を防止できる。

サーチエンジンボットによるインデックス化の防止: サーチエンジンによるインデックス化の抑制は HTML タグで可能だが、その機能は完全には保証されていない。CAPTCHA による認証後にページの読み込みを許可することで、サーチエンジンによるページのインデックス化を防止できる。

本稿では、 Ahn[36] らの数学的な定義に従い、CAPTCHA で使用する問題として、AI (Artificial Intelligence) 問題を用いる。人間とロボットの判別は、この問題の解答結果によっておこなう (図 1.1)。代表的な AI 問題には、歪んだ文字画像の解釈 (図 1.2[5]) や、 変形した音声の聞き取りがある。現在普及している CAPTCHA の多くは、使用する AI 問題の種類により、画像型と音声型に大別される。

1.2 研究の背景

現在利用されている CAPTCHA には、次のような問題が指摘されている。

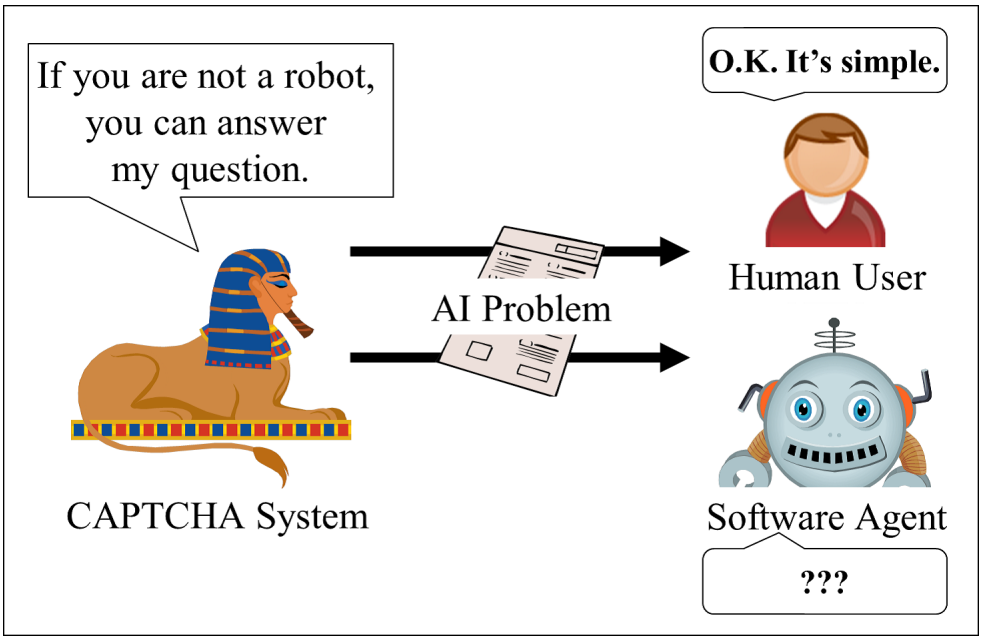


Fig. 1.1　Aspect of a CAPTCHA.

Fig. 1.2　Example of a CAPTCHA.

アクセシビリティ

AI 問題には、特定知覚の解釈能力を利用するものが多い。画像型 CAPTCHA であれば視覚を、音声型 CAPTCHA であれば聴覚を利用し、人間と現在のロボットが備える解釈能力の差を判別の根拠としている。そのため CAPTCHA は、知覚に障害を持つ利用者のウェブアクセシビリティを阻害する障壁となっている。

ガイドラインである WCAG2.0[8] では、CAPTCHA に対して、特定知覚に障害を持つ利用者のため、代替知覚を用いた方式を併用することを要求している。しかしながら、表 1.1 に示されるように、音声型 CAPTCHA は十分に普及していない。

テストの難化

CAPTCHA は、 2000 年に発明されてより急速に普及しているが、 その一方で多くの攻撃にさらされてきた。近年では、計算機の性能向上やアルゴリズムの改良により、ロボットによる攻撃の成功例が多数報告されている。

初期の画像型 CAPTCHA は、OCR (Optical Character Reader) を用いた Yan ら [39] の攻撃や SVM

Table 1.1 Service Appearance of CAPTCHAs in November 2013.

Service Name	Visual CAPTCHA	Audio CAPTCHA	Other Type of CAPTCHAs	Telephone Dialogue
Microsoft	x	x		
Google	x	x		x
Yahoo! Japan	x	x		
Amazon	x	x		x
WordPress	x			
Ameba Blog	x			
F2C Blog	x			
White House Petition			x†	

†: The type is quiz.

(Support Vector Machine) を用いた Tam ら [35] の攻撃によって破られ、その度に CAPTCHA 側は、テストの難易度を上げて対抗してきた。攻撃側も改善を重ね、Bursztein ら [14] が当時の商用サービスで利用されていた大部分の 画像型 CAPTCHA を破ると、その難化傾向がより顕著になった。

音声型 CAPTCHA は、人間にも難しいと指摘 [12, 13, 34] されている。特に、機械学習を用いた方式 [6] により音声型 CAPTCHA が破られてからは、もはや人間にもほとんど解けないほどに、テストは難化してしまった。

この状況において、特に問題となるのは、視覚障害者への影響である。現在の音声型 CAPTCHA は、普及が十分でない上に、例え導入されていたとしても、人間にも解答不能なほどに難しい。よって、現在普及している CAPTCHA を用いたサービスは、視覚障害者にとって利用が困難な状況にある。2013 年には、NFB (National Federation of the Blind) やオンライン請求サイト「Change.org」において、CAPTCHA のアクセシビリティ問題が相次いで指摘 [1] されている。

1.3 研究の目的

実社会で使用される CAPTCHA の多くは、特定知覚の使用を強制するため、その知覚に障害がある利用者の障壁となっている。本研究の目的は、この問題を解消し、知覚障害によらず利用可能な、定義 1 を満たす CAPTCHA の作成である。

定義 1. (バリアフリーな CAPTCHA) バリアフリーな CAPTCHA とは、次に示す全ての要件を満たす。

バリアフリー要件: 特定の知覚のみの使用に限定されない。

知識非依存性要件: テストの難易度が特定の知識の有無に強く依存しない。

[1] White House により提供されているオンライン請願「We the People」は、この指摘によりクイズ形式の CAPTCHA を用いている。しかしこの方式は、質問形式の種類が少ないため、簡単な単語辞書と質問文のマッチングによって、高い確率でロボットに解答できることが推測される。

識別性要件: 人間には容易に解けるが、現状のロボットには解答が難しい問題を生成できる。

問題新規性要件: 未使用で新規な問題を無数かつ自動で作れる。 ■

1.4 研究のアプローチ

定義 1 の要件を満たす CAPTCHA を構成するためのアプローチを示す。

- 提案方式は、AI 問題として、文意や文脈を解釈する問題（文意文脈解釈問題と称す）を用いる。この種の問題は、その困難性が特定知覚に依存しないため、バリアフリー要件を満たす。
- 提案方式は、文意文脈解釈問題の素材となる文章（素材文章）に、インターネット上の公開文章を用いる。これらは豊富な分量を持つため、素材文章を更新しながら作問することで、問題新規性要件を満たす。既存研究の多くは、分量の少ない秘匿文章を用いているため、この点に関する検討が不足している。
- 公開文章から生成した問題には、攻撃者が検索により、解答のヒントとなる情報を引き抜く危険がある。提案方式は、問題文の子音を改変（子音交替と称す）することで、検索による攻撃を妨害する。攻撃者が改変結果から元の文を推測することは、その候補数が膨大であるため困難である。

提案方式は、具体的な文意文脈解釈問題として、(a) 機械合成文と自然文の識別問題、(b) 機械翻訳文と自然文の識別問題、(c) 文章に共通する話題の識別問題、を用いる。(a) や (b) は、文章の自然さの判別であり、特定知識を強くは要求しない。(c) については、一般的な話題を選択することで、特定知識への依存を避けることができる。したがって、提案方式は、知識非依存性要件を満たす。

子音交替によって改変された文は、誤植や聞き間違いといった「間違い」を含んだ文に似ている。提案方式では、人間の「間違い」を修復する能力や文脈解釈能力により、人間には元の文意を解釈できることを期待している。一方、ロボットには、文意文脈解釈問題を解くこと[2]や、公開文章から解答のヒントを取得するこが難しい。したがって、提案方式は、識別性要件を満たす。

1.5 本研究の貢献

本研究では、定義 1 に示した要件を満たす CAPTCHA システムを構成した。

まず、前述のアプローチに基づき、文意文脈解釈問題のアルゴリズムを提案した。本研究では、ウェブ上の公開文章から生成した問題文に対して子音交替を施すことで、問題新規性要件を満たしつつ、ロボットによる解答を困難にしている。

次に、評価プログラムを実装し、提案方式による作問結果が人間の解ける問題であるかどうかを、視覚障害者を被験者とした実験により確認した。提案方式は、既存の Google 音声型 CAPTCHA に比べ、高い確率で人間を「ロボットではない」と判別できている。さらに、ロボットによる攻撃方法を分析し、提案方式の安全性の検討をおこなった。これらの評価・考察により、提案方式が有用であることを示した。

[2](a) から (c) に示した文意文脈解釈問題には、いくつかの攻撃方法が想定される。本研究では、その対策も講じている。詳細は、3 章以降を参照されたい。

1.6　本稿の構成

1 章では、AI 問題の知覚依存という CAPTCHA のアクセシビリティ問題を取り上げ、解決のためのアプローチと本研究の貢献について述べた。2 章では、3 章以降の説明に必要となる表記や用語について、本稿で使用する定義とその概要を示す。3 章では、言語的作問技法を用いた CAPTCHA システムの基本構成を示し、具体的な AI 問題として 3 種類の文意文脈解釈問題を取り上げ、それぞれの構成法を示す。4 章では、3 章で提案した方式について、その評価・考察を示す。また、今後の課題となる内容について、議論をおこなう。5 章は、本稿のまとめである。

2 諸定義

本章では、次章以降で用いる諸定義を述べる。

2.1 節では表記、2.2 節では自然言語処理に属するいくつかの操作、2.3 節では n 階マルコフ連鎖、2.4 節ではワードサラダ、2.5 節では BLEU 、2.6 節では統計的有意性、そして 2.7 節では CAPTCHA について、各々の概要とその定義を振り返る。

2.1 表記

オブジェクトの集まりを表現するデータ構造をコンテナと称す。オブジェクトには変数やリテラルがあり、コンテナには配列、連想配列、集合などがある。コンテナ内の要素の重複度を考慮する場合は、多重コンテナとして扱う。例えば多重集合では、$\{a, b\}, \{a, a, b\}$ を異なる集合として扱う。多重連想配列の場合は、連想鍵と連想値の両方で重複度を考慮する。本稿では、集合に対して使用する演算記号を、コンテナに対しても適用する。例えば配列 $(1, \ldots, 9)$ の任意の要素 x は、$x \in (1, \ldots, 9)$ のように表す。

本稿では、$y \leftarrow x$ と書くとき、x がコンテナであればそこから要素を取り出し y に代入、x が演算であれば結果を y に代入、x がアルゴリズムまたは関数であれば y を出力する操作を表す。これらの操作がある確率分布 D に従うことを明記する場合は、$y \xleftarrow{D} x$ のように表す。例えば、$A = \{(key, val_0), (key, val_0), (key, val_0), (key, val_1)\}$ となる多重連想配列を考えると、$val \xleftarrow{D} A(key)$ の操作により、val に 75% の確率で val_0、25% の確率で val_1 が代入される。$D = \$$ の場合は、一様ランダムな分布に従うことを表す。

$|z|$ は、z がコンテナであれば要素数を、オブジェクトであればそのデータ型に応じた値を表す。例えば、文字列型であればその文字列長、整数型であればそのビット長を表す。$a||b$ は、a と b の連結を表す。

2.2 自然言語処理

2.2.1 形態素解析

形態素解析 (Morphological Analysis) とは、自然言語で記述された文を形態素 (Morpheme) に分割し、それぞれの品詞を判別する操作である。形態素は、おおまかにいえば、言語で意味を持つ最小の単位である。

本稿では、品詞などの情報付き単語リストを辞書に持ち、対象言語の文法知識と合わせて解析をおこなう、教師ありの方式を扱う。具体的なツールとして、MeCab[24] を用いる。解析例を、表 2.1 に示す。

2.2.2 構文解析

構文解析 (Syntactic Analysis) とは、ある文章の文法的な関係を調べる操作である。日本語の構文解析では、文節間の係り受け構造を調べることに相当する。

本稿では、具体的なツールとして Cabocha[25] を用いる。例として「私は夕食を食べた」を解析すれば、「私は → 食べた」、「夕食を → 食べた」となる係り受け構造を得る。

Table 2.1　Example of Morphological Analysis.

Text	Pos	Pos1	Pos2	Pos3	InflectedType	InflectedForms	Read	Pronunciation
私	名詞	代名詞	一般	*	*	*	ワタシ	ワタシ
は	助詞	係助詞	*	*	*	*	ハ	ワ
夕食	名詞	一般	*	*	*	*	ユウショク	ユーショク
を	助詞	格助詞	一般	*	*	*	ヲ	ヲ
食べ	動詞	自立	*	*	一段	連用形	タベ	タベ
た	助動詞	*	*	*	特殊・タ	基本形	タ	タ

Notation: The first row means properties of default database in MeCab.

2.2.3　コーパス

コーパスとは、自然言語処理で用いられる、自然言語の文章を構造化し集積したデータベースである。例としては、n 個の形態素や文字の順序付き組み合わせを集積したものがある。このように、ある n 個のデータ型 X の順序付き組み合わせを X n-gram 、それを集積したものを X n-gram データベースと称す。

コーパスの生成には、自然言語で記載された素材となる文章が必要になる。さらに、素材文章を収集するには、その収集先となる文章（文章源と称す）が必要になる。コーパスの生成の例としては、新聞記事を文章源とし、そこから日本語の文を素材文章として取得し、その形態素解析結果を形態素 n-gram ごとに集積する方法がある。

利用価値が高いコーパスは、集積されるデータの要素に対して、その出現頻度や、品詞などの言語的な情報がタグとして付与される。本稿で使用するコーパスでは、品詞などの言語的な情報は MeCab に同伴されている辞書を、出現頻度や係り受け関係などは収集した素材文章の解析結果を利用する。

2.2.4　カルバック・ライブラー・ダイバージェンス

カルバック・ライブラー・ダイバージェンス [20] （Kullback-Leibler divergence: KL ダイバージェンスと称す）とは、2 つの確率分布に対して、それらの間の異なり具合を測るものである。確率分布 P, Q が与えられたとき、P からみた Q の KL ダイバージェンス $D_{KL}(P\|Q)$ は、式 (2.1) のようになる。

$$D_{KL}(P\|Q) = \sum_{x} P(X = x)\log\frac{P(X = x)}{Q(X = x)} \tag{2.1}$$

言語処理においては、単語間の意味的な遠さを測るため、KL ダイバージェンスを用いる。KL ダイバージェンスは、その値が小さいほど、意味的に近い関係を表すことに注意を要する。

2.3　n 階マルコフ連鎖

n 階マルコフ連鎖は、直前 n 個の状態に依存して次の状態が決定される確率過程である。これは、式 (2.2) のように定式化される。また、n を階数と称す。

$$\begin{aligned} & P(X_{i+1} = x|X_i = x_i, \ldots, X_0 = x_0) \\ = \quad & P(X_{i+1} = x|X_i = x_i, \ldots, X_{i-n+1} = x_{i-n+1}) \end{aligned} \tag{2.2}$$

2.4 ワードサラダ

本研究では、ワードサラダと呼ばれる機械合成文を用いる。ワードサラダは、文中のある形態素が直前の *n*-gram 形態素のみにより決まる連鎖型共起表現から構成される。このためワードサラダは、「てにをは」といった文法構造はある程度正しいが、登場する単語はある確率分布に従いランダムに選ばれる。その内容は、ランダム性に応じて不自然な文になる。

ワードサラダは、文法に大きな誤りはないため、文法チェッカなどのプログラムでは、 人間が作る自然文と見分けがつかない。よって、機械合成文と自然文の識別には、文章の内容についての「自然さ」を評価する必要がある。これには常識が必要となるため、ロボットには解答困難だと期待できる。

ワードサラダ生成には、コーパスとして形態素 *n*-gram データベースを、言語モデルとして式 (2.2) に示したマルコフ連鎖をしばしば用いる。単純な生成プログラムでは、コーパスから形態素 *n*-gram を取り出し、素材文章から導出したマルコフ連鎖に従い連鎖する形態素を再帰的に選択することで、ワードサラダを生成できる。

ワードサラダのランダム性は、生成に使用した階数 n に依存する。マルコフ連鎖では、連続した n 個の形態素の組み合わせに対して、$n+1$ 番目の形態素が選択される。n が大きければ、 $n+1$ 番目の形態素を選択するのに必要な条件が厳しくなる。したがって、n が大きいほど、生成される文のランダム性は小さく、自然な文になる。n を小さくすれば、生成される文のランダム性は高く、不自然な文になる。

2.5 BLEU

BLEU[16] (BiLingual Evaluation Understudy) とは、機械翻訳 (MT) の自動評価尺度の 1 種である。参照訳と MT 訳における形態素 *n*-gram の一致度により計算されるスコアは、0.0–1.0 の範囲の値を取り、大きいほどよい MT 訳とされる。本稿では、BLEU を 2 つの文の類似度を表す指標としても利用する。

K 個の参照訳と MT 訳を、それぞれ $\mathcal{E} = \{e_1, \ldots, e_K\}, \hat{\mathcal{E}} = \{\hat{e}_1, \ldots, \hat{e}_K\}$ とする。各要素は訳文であり、1-gram 形態素を要素とする順序付き配列で構成されているとする。MT 訳 $\hat{e}_k$ に対して *n*-gram 数を計算する関数 $c_n(\hat{e}_k)$ は、式 (2.3) のようになる。

$$c_n(\hat{e}_k) = |\hat{e}_k| - n + 1 \tag{2.3}$$

訳文の組 $(e_k, \hat{e}_k)$ に対して *n*-gram の一致数を計算する関数 $m_n(e_k, \hat{e}_k)$ は、式 (2.4) のようになる。ここで x は、$\hat{e}_k$ の中に含まれる任意の *n*-gram 形態素を表す。$a(\cdot, x)$ は、第 1 引数に訳文を取り x との *n*-gram 一致数を返す関数を表す。

$$m_n(e_k, \hat{e}_k) = \sum_x \min(a(\hat{e}_k, x), a(e_k, x)) \tag{2.4}$$

n-gram の適合率 $P_n(\mathcal{E}_k, \hat{\mathcal{E}}_k)$ は、式 (2.5) のように計算する。

$$P_N(\mathcal{E}, \hat{\mathcal{E}}) = \frac{\sum_{k=1}^{K} m_n(e_k, \hat{e}_k)}{\sum_{k=1}^{K} c_n(\hat{e}_k)} \tag{2.5}$$

参照訳に対して短い MT 訳には簡易ペナルティ $\mathrm{BP}(\mathcal{E}, \hat{\mathcal{E}})$ を、式 (2.6) のように与える。なお、このペナルティを Breavity Penalty[31] と称す。

$$\mathrm{BP}(\mathcal{E}, \hat{\mathcal{E}}) = \begin{cases} 1 & (\text{if } |\hat{\mathcal{E}}| > |\mathcal{E}|) \\ \exp\left(1 - \frac{|\mathcal{E}|}{|\hat{\mathcal{E}}|}\right) & (\text{otherwise}) \end{cases} \tag{2.6}$$

一般的には、n = 1–4 の値を組み合わせることで式 (2.7) のように BLEU を計算する。

$$\mathrm{BLEU}(\mathcal{E}, \hat{\mathcal{E}}) = \mathrm{BP}(\mathcal{E}, \hat{\mathcal{E}}) \times \exp\left(\sum_{n=1}^{4} \frac{\log P_n(\mathcal{E}, \hat{\mathcal{E}})}{4}\right) \tag{2.7}$$

BLEU+1

BLEU は、広く利用されている評価尺度ではあるが、訳文ごとへの適用はできない。なぜなら、$n = 3, 4$ などの大きい n で訳文ごとに評価すると、式 (2.5) の値が 0 になりやすく正しく評価できないためである。そこで、式 (2.5, 2.7) を式 (2.8, 2.9) のように改良した BLEU+1[28] が提案されている。

$$P'_N(\mathcal{E}, \hat{\mathcal{E}}) = \begin{cases} \left\{\sum_{k=1}^{K} m_n(e_k, \hat{e}_k)\right\} \Big/ \left\{\sum_{k=1}^{K} c_n(\hat{e}_k)\right\} & (\text{if } n = 1) \\ \left\{\sum_{k=1}^{K} m_n(e_k, \hat{e}_k) + 1\right\} \Big/ \left\{\sum_{k=1}^{K} c_n(\hat{e}_k) + 1\right\} & (\text{otherwise}) \end{cases} \tag{2.8}$$

$$\mathrm{BLEU} + 1(\mathcal{E}, \hat{\mathcal{E}}) = \mathrm{BP}(\mathcal{E}, \hat{\mathcal{E}}) \times \exp\left(\sum_{n=1}^{4} \frac{\log P'_n(\mathcal{E}, \hat{\mathcal{E}})}{4}\right) \tag{2.9}$$

スムージング BLEU+1

BLEU+1 の計算は、Breavity Penalty については BLEU と相違ないため、文ごとの BLEU+1 から最適解を計算する場合、短文に関してはバイアスがかかる問題がある。修正案としてスムージング型の BLEU+1 が提案 [29] されている。本研究では短文について評価するため、これを指標として利用する。スムージング BLEU+1 では、Breavity Penalty を式 (2.10) のように与える。

$$\mathrm{BP}(\mathcal{E}, \hat{\mathcal{E}}) = \begin{cases} 1 & (\text{if } |\hat{\mathcal{E}}| > |\mathcal{E}|) \\ \exp\left(1 - \frac{|\mathcal{E}|+1}{|\hat{\mathcal{E}}|}\right) & (\text{otherwise}) \end{cases} \tag{2.10}$$

2.6 統計的有意性

本研究では、実験結果を考察する際に、独立する 2 群の有意性を検定するときは、ノンパラメトリックな検定方法である Mann-Whitney の U 検定（Mann-Whitney U Test）を使用する。また、1 つの群で異なる 2 つの実験条件での有意性を検定するときは、次に示す方法を使用する。

- 有意水準は、心理実験で慣用である 5% とする。
- 被験者群の人数を N とする。被験者 $i \in \{0, \ldots, N-1\}$ ごとに、実験条件 1 の標本 $A_{1,i}$ と実験条件 2 に対する標本 $A_{2,i}$ を調べる。$A_{1,i} > A_{2,i}$ ならば $D_i = 1$、$A_{1,i} < A_{2,i}$ ならば $D_i = -1$、$A_{1,i} = A_{2,i}$ ならば $D_i = 0$ とする。D_i について、N 人の被験者について和を求め、$D = \Sigma_{i=0}^{N-1} D_i$ を得る。
- 帰無仮説として、D_i が正負ランダム、すなわち二項分布すると仮定する。
- 二項分布に従い D の値を調べ、$|D| \geq X$ となる確率が 5% 以下となる X を探索する。X が存在すれば、有意水準 5% を満たすので、帰無仮説を棄却する。すなわち、有意差ありと判定する。

2.7 CAPTCHA

AI 問題

本節では、Ahn ら [36] により示された CAPTCHA の数学的な定義を紹介する。セキュリティパラメータの取り扱い方が通常の暗号理論のものと異なる点に注意を要する。さらに、本稿では、問題文の文章量を考慮してた定義に拡張する。

CAPTCHA システムには、 検証者（自動プログラム）と証明者（人間、またはロボット）のエンティティが存在する。CAPTCHA は、検証者と証明者の対話によるチャレンジ／レスポンス型テストである。検証者は、AI 問題を作問し証明者に提示する。さらに、証明者からの解答を照合し、人間とロボットの判別をおこなう。検証者を $\mathcal{V}$ とし、人間を $\mathcal{P}$、ロボット（すなわち、攻撃者）を $\mathcal{P}^*$ とする。

κ を AI 問題のセキュリティパラメータとし、$\ell_0(\cdots), \ell_1(\cdots)$ を多項式とする。本稿では言語的作問を扱うので、これらは問題文の文章量に影響を与える。作問に使用する文字集合を C とする。κ を入力とした確率的多項式時間（Probabilistic Polynomial Time, PPT と称す）な作問アルゴリズム $\mathcal{G}$ により生成される問題の空間を $\mathcal{T}_k \subseteq C^{\ell_0(\kappa,\ldots)}, \mathcal{T} := \{\mathcal{T}_k\}_{k\in\mathbb{N}}$ とし、その回答の空間を $\mathcal{S}_k \subseteq C^{\ell_1(\kappa,\ldots)}, \mathcal{S} := \{\mathcal{S}_k\}_{k\in\mathbb{N}}$ と表す。$\mathcal{H} := \{H_k\}_{k\in\mathbb{N}}$ は、$H_k : \mathcal{T}_k \to \mathcal{S}_k$ となる写像とする。AI 問題は $\mathcal{Q} = (\mathcal{G}, \mathcal{H})$ で定義され、$H_k(z) = a$ となる問題と回答のペア $(z, a) \in \mathcal{T}_k \times \mathcal{S}_k$ で表される。

アルゴリズム $\mathcal{A}$ が、AI 問題 $\mathcal{Q}$ を解く確率 $\mathrm{Adv}_{\mathcal{A}}^{\mathcal{Q}}(\kappa)$ を式 (2.11) のように定義する。表記の簡略化のため、以降 $\epsilon_{\mathcal{A}} := \mathrm{Adv}_{\mathcal{A}}^{\mathcal{Q}}(\kappa)$ とする。

$$\mathrm{Adv}_{\mathcal{A}}^{\mathcal{Q}}(\kappa) := \Pr\left[\begin{array}{r}(z, a) \leftarrow \mathcal{G}(\kappa, \ldots); \\ a' \leftarrow \mathcal{A}(\kappa, z, \ldots); a = a';\end{array}\right] \tag{2.11}$$

高々 $\tau_{\mathcal{A}}$ の実行時間により AI 問題 $\mathcal{Q}$ を少なくとも $\epsilon_{\mathcal{A}}$ 以上の確率で解くアルゴリズムを $(\epsilon_{\mathcal{A}}, \tau_{\mathcal{A}})$-$\mathcal{A}$ と称する。このときアルゴリズム $\mathcal{A}$ は、$\mathcal{G}$ で使用される内部乱数以外の全ての知識を持つことに注意を要する。

$(\epsilon_{\mathcal{P}^*}, \tau_{\mathcal{P}^*})$-$\mathcal{P}^*$ となるいかなるロボットも存在しない場合は、$\mathcal{Q}$ を、ロボットにとって $(\epsilon_{\mathcal{P}^*}, \tau_{\mathcal{P}^*})$-困難な AI 問題と称す。一方、システムの対象となる人間 $\mathcal{P}$ の多くが、高々 $\tau_{\mathcal{P}}$ の時間と少なくとも $\epsilon_{\mathcal{P}}$ の確率で AI 問題を解ける場合は、$\mathcal{Q}$ を、人間にとって $(\epsilon_{\mathcal{P}}, \tau_{\mathcal{P}})$-容易な AI 問題と称す。表記の簡略化のため、文脈上明らかな場合は、「ロボットにとって」や「人間にとって」の記述を省略する。

CAPTCHA システム

AI 問題により CAPTCHA システムを構成するには、式 (2.12) の条件を満たす必要がある。ただし、$\hat{\tau}_{\mathcal{P}}$ はシステムが許容する問題文提示から解答までの遅延時間の上限値、$\hat{\epsilon}_{\mathcal{P}}$ はシステムが許容する $\mathcal{P}$ による正答確率の下限値、$\hat{\epsilon}_{\mathcal{P}^*}$ はシステムが許容する $\mathcal{P}^*$ による攻撃成功確率の上限値とする。ただし、$\mathcal{P}$ に関するパラメータは、システムの対象となる人間の言語や知識により変化する。本稿では、標準的な日本人の成人を対象とする。

$$\begin{aligned}
&\tau_{\mathcal{P}}(\kappa) \leq \hat{\tau}_{\mathcal{P}}(\kappa) \\
&\epsilon_{\mathcal{P}}(\kappa) \geq \hat{\epsilon}_{\mathcal{P}}(\kappa) \\
&\epsilon_{\mathcal{P}}(\kappa) \gg \epsilon_{\mathcal{P}^*}(\kappa) \qquad \text{if } \tau_{\mathcal{P}^*}(\kappa) \leq \hat{\tau}(\kappa). \\
&\epsilon_{\mathcal{P}^*}(\kappa) < \hat{\epsilon}_{\mathcal{P}^*}(\kappa) \qquad \text{if } \tau_{\mathcal{P}^*}(\kappa) \leq \hat{\tau}_{\mathcal{P}}(\kappa).
\end{aligned} \tag{2.12}$$

AI 問題を解く確率が人間とロボットの間で乖離がある場合、AI 問題を繰り返し解くことでその乖離を大きくできる。これを利用した CAPTCHA システムは、n 回の出題のに対し k 回以上の正答によって「ロボットではない」と判別する。これ以降、本稿では式 (2.12) を、繰り返しを考慮した値として取り扱う。すなわち n, k は、セキュリティパラメータの 1 種とする。

3　言語的作問技法を用いた CAPTCHA システム

本章では、提案方式である文意文脈解釈問題を用いた CAPTCHA システムについて述べる。

3.1 節では、文意文脈解釈問題を用いた CAPTCHA システムの基本構成を示す。3.2 節において、提案内容の 1 つである公開文章の利用の理由を明らかにし、3.3 節にて、公開文章を安全に利用するための方法を提案する。3.4 節では、文意文脈解釈問題の詳細を示す。

3.1　文意文脈解釈問題を用いた CAPTCHA システムの基本構成

本稿で想定する運用モデルは、エンティティとして、サービス提供者とその利用者を考える。サービス提供者、すなわち CAPTCHA システムを検証者 $\mathcal{V}$ とする。サービスの利用者を証明者 $\mathcal{P}$ とする。提案する CAPTCHA システムは、$\mathcal{V}$ と $\mathcal{P}$ の対話により、大別して次の手順をたどる。

(1) 作問プログラムの起動: $\mathcal{V}$ は、対話中である $\mathcal{P}$ が「ロボットではない」ことを確認する必要が生じると、文意文脈解釈問題の作問プログラムを起動する。なお、$\mathcal{V}$ と作問プログラムは、どちらもソフトウェアであることに注意を要する。

(2) コーパスの準備: 作問プログラムはインターネットに接続し、問題文の生成に必要な素材文章を収集する。さらに、収集した素材文章を解析し、コーパスとして保存する。

一般に、コーパスの生成にはある程度の時間を要する。よって、本研究では、あらかじめ適当な分量のコーパスを生成しておき、必要に応じて更新することを考える。

(3) 作問: $\mathcal{V}$ は、コーパスを用いて、複数の文を問題文 z として作成する。このうち特定の文だけに、「内容の自然さ」や「話題分野の一致」といった文意解釈上の特徴が含まれているようにし、それを解答 a として保存する。

(4) 出題: $\mathcal{P}$ に、作成した問題文 z を提示し、特定の特徴を持った文を選ぶようにと指示する。問題は文字情報として提示されれば十分なので、視覚／点字ディスプレイや音声読み上げのいずれの手段にも対応でき、バリアフリー化が果たされる。

(5) 解答の受付: $\mathcal{V}$ は、出題に合わせて内部タイマを初期化・起動し、$\mathcal{P}$ からの解答 a' が来るまで待機する。待機中に内部タイマの値が $\hat{\tau}_{\mathcal{P}}$ を超えた場合には、タイムアウトフラグを立てる。そうでなければ、$\mathcal{P}$ の解答をトリガとして、内部タイマを停止する。

(6) 解答の照合: $\mathcal{V}$ は、タイムアウトフラグが立っている場合は、フラグをクリアし a' を不正解として扱う。そうでなければ、$a = a'$ であれば a' を正解とし、$a \neq a'$ であれば不正解とする。

(7) 認証: $\mathcal{V}$ は、(2)–(6) の処理を n 回繰り返す。$\mathcal{P}$ の正解数が k 回以上ならば、「ロボットではない」と認証する。

n, k の値は、システムの要求仕様となる式 (2.12) の値を満たすように決定する。

3.2 文章源

文意文脈解釈問題の作成には、素材文章を解析して構築したコーパスが必要になる。本節では、素材文章の取得先である文章の集合、すなわち文章源について検討する。文章源の選択と、そこから素材文章を獲得する方法には、次のものが考えられる。

(1) インターネット から文章を収集する方法: Twitter やブログなどで公開された文章の集合を文章源とし、そこから素材文章を収集する。多様な文章を大量に得られる利点がある。反面、それは公開情報であるから、攻撃者が同じ文章を検索によって収集し、解答のヒントを得る恐れがある。それゆえ文章源と同じ文章を、改変せずにそのまま解答の選択肢に使う方式 [23, 41, 45] には特に不向きである。

(2) 秘匿文章を利用する方法: 出題者だけが知り得る文章の集合を文章源とし、そこから素材文章を収集する。公開文章ではないので、攻撃者が、検索により直接情報を得ることはできない。反面、秘匿文章は分量が少ないため、公開文章ほど低コストかつ大量に素材文章を収集できない。

(3) 証明者に文章を作成・入力させる方法: テストの途中に、証明者が文章を作成・入力する段取りを持ち、その結果を文章源とする。これにより、新規な文書がテストシステムに補給され、別の機会に素材として使用できる。ただし、テスト手順が煩雑になる問題がある。

(2)、(3) の方式は、検索による直接的な攻撃はできないため、(1) の方式より安全に見える。しかし、そうではないことを本稿では指摘する。

(2) のように秘匿文章を用いた場合でも、問題新規性要件を満たさない場合には安全とはいえない。なぜなら、1 度問題として使えば、そこで公開されてしまうので、秘匿が成り立たない。攻撃者は、テストシステムへの出題要求を大量に繰り返すことで、問題文と解答についてのヒントを得られる。テストシステムが問題新規性要件を満たさなければ、攻撃者は、いつかは既知で解答を分析済みの問題に遭遇する。残念ながら、秘匿文章は分量の問題があるため、問題新規性要件を満たさない。

(3) については、攻撃者も文章生成に参加できるため、攻撃者に都合の良い素材文章を登録できてしまう。また、(2) と同様の問題も存在する。

よって本研究では、問題新規性要件を満たす方式を実現するため、(1) の方式を採用する。

すでに述べたように、(1) の方式は公開文章を利用する。このままだと、検索により問題に対するヒントが得られるため、ロボットを否認する点で識別性要件を満たさない。3.3 節では、この対策について示す。

3.3 子音交替

文章源の特定を妨害するため、提案方式では、問題文として提示する文字列の子音を改変する。これは、方言などに見られる単語の子音の違いを指す。例えば、ザ行からダ行への子音交替では、「“ざ” ぶとん」を「“だ” ぶとん」と改変する。漢字に対しては、CABOCHA[25] の「読み」情報を利用し仮名に開いてから、子音の改変をおこなう。図 3.1 の 2 例は、子音交替を 2 箇所適用したものと 3 箇所適用したものである。

- 七時過ぎに夕食を食べた。
  →（仮名開き）ななじすぎにゆうしょくをたべた。
  →（子音改変）ななじす じ にゆう ひょくをたべた。
- お金が足りず本が買えない。
  →（仮名開き）おかねがたりずほんがかえない。
  →（子音改変）お が ねがたりずほ ろ が が えない。

Fig. 3.1　Examples of Applying a Process of Consonant Gradation.

子音交替の手順を説明するため、使用する記号について示す。子音交替前の文字列を s_p とし、漢字を仮名に開く関数を openCC とする。日本語における「ア行、カ行、･･･」など、全ての行の種類の集合を $\mathcal{U}_L$ とする。仮名に開いた文字列 s_m に含まれる、すべての行の種類の順序付き多重配列を $\mathcal{U}_{s_m}$ とする。例えば、s_m = “あさがお” であれば、$\mathcal{U}_{s_m}$ = (“ア行”, “サ行”, “ガ行”, “ア行”) となる。

子音交替関数は、s_p と改変箇所の上・下限値 r_L, r_H $(r_H > r_L > 0)$ を入力とし、$\mathrm{cogd}(s_p, r_L, r_H)$ と表す。処理内容を、次に示す。

(1) 改変箇所数 r を、$r \xleftarrow{\$} \{r_L, \ldots, r_H\}$ と取得する。仮名に開いた文字列 s_m を、$s_m \leftarrow \mathrm{openCC}(s_p)$ と取得する。

(2) s_m から $\mathcal{U}_{s_m}$ を計算する。$|\mathcal{U}_{s_m}| < r$ であれば、$r \leftarrow |\mathcal{U}_{s_m}| - 1$ とする。

(3) $r > 0$ であれば、(4) に進む。そうでなければ、s_m を出力し、処理を終了する。

(4) u, v を次のように選択する。このとき、u が $\mathcal{U}_{s_m}$ 中に格納されていた順番を、i とする。

$$u \xleftarrow{\$} \mathcal{U}_{s_m}, v \xleftarrow{\$} \mathcal{U}_L \backslash u$$

(5) s_m の i 番目の要素 u を v に置換した文字列 s'_m を取得する。$s'_m \neq s_m$ であれば、$r \leftarrow r - 1$、$s_m \leftarrow s'_m$ とする。

(6) (2) に進む。

r_L, r_H の値は、識別性要件を考慮して決める必要がある。人間は、改変箇所が少なければ元の文章と同様の意味を理解できる [32, 33] が、改変箇所が増えるにつれ文脈解釈が難しくなる。ロボットが改変前の文字列を求める場合、改変箇所が増えるにつれ、その候補となる数は爆発的に増加する。したがって、改変箇所数は、ロボットが $\hat{\tau}_{\mathcal{P}}$ (2.7 節参照) の時間以内に、元の文字列の推測が計算困難である最小値を選択する。

実際にロボットが提案方式を攻撃をするには、さらにいくつか困難な問題を解く必要がある。まず、ロボットには、子音改変前の文字列 s_m の知識がないため、復元した文字列が正しいかを確実には判定できない。次に、子音交替を適用した文字列は仮名に開かれているため、ロボットは文章源を

直接検索できない。文章源の文字列を仮名に開いたデータベースがあればよいが、提案方式ではインターネット上の文章が対象になるため、現実的でない。s_m を漢字仮名交じりの文字列に変換してから文章源を検索する手法もあるが、ロボットは s_p の知識がないので、正しい漢字変換を選択できない。この場合、子音交替により形態素解析が妨害され、漢字への変換対象を正しく選択できないことも問題となる。

3.4　文意文脈解釈問題

本節では、文意文脈解釈問題として、ワードサラダ文識別テスト、機械翻訳文識別テスト、そして共通話題識別テストの構成を示す。はじめに、全てのテストに共通して必要となる自然文の生成方法、すなわち、素材文章からの文字列の切り出し方について述べる。次に、各テストのアルゴリズムや作問例を示す。

3.4.1　自然文の生成方法

文意文脈解釈問題では、人間の作成した文章を利用して、作問することが多々ある。本節では、人間の作成した文章を、素材文章からどのように切り出すかを説明する。

最も単純なアイディアは、句点を基準にして文字列を分割し、それらを利用することである。しかしながら、素材文章の切り出しをこのような文単位でおこなうと、作問で生成される文の種類が制限されてしまう。本研究ではインターネット上の多量な文章を利用するが、文単位では数に制限のない作問はできない。句読点を基準に文字列の分割をすれば若干の改善は図れるが、それでも十分とは明言できない。

本研究では、この問題を回避するため、素材文章からの文字列切り出し関数 nsgen を提案する。この関数は、改行区切りの文章[1] m から、文字数の範囲が len_L–len_H となる文字列を出力する。ただし、$|m| > len_H > len_L > 0$ とする。nsgen の処理内容を、次に示す。

(1) 形態素解析により m の形態素配列 mor を取得する。mor の i 番目の形態素を $mor[i]$ とする。mor のサイズを +1 し、追加要素として、終端を表す特殊記号（終端記号と称す）を配列の最後に格納する。

(2) $\mathcal{S}$ を、文字列を要素とする集合とする。$i = 0, \mathcal{S} \leftarrow \emptyset$ とし、次の処理をおこなう。

(2-1) 出力する文字列長の下限値を $len \overset{\$}{\leftarrow} \{len_L, \ldots, len_H\}$ と決定する。

(2-2) $i < |mor|$ ならば、 (2-3) に進む。そうでなければ、(3) に進む。

(2-3) $mor[i]$ が自立語でなければ、$i \leftarrow i+1$ とし、(2-2) に進む。$mor[i]$ が自立語であれば、$mor[i]$ から総文字列長が len 以上となる配列要素までの形態素を結合し、文字列 s の生成を試みる。処理の途中で終端記号を読みだした場合は、(3) に進む。そうでなければ、(2-4) に進む。

文字列の切り出しを自立語からおこなう理由は、なるべく自然な形で文字列を切り出すためである。

[1]MS-WORD のパラグラフに相当し、複数の文を許容する。

(2-4) $|s| \leq len_H$ であれば、$\mathcal{S} \leftarrow \{\mathcal{S}, s\}$ とする。$i \leftarrow i+1$ とし、(2-1) に進む。

(3) $s' \overset{\$}{\leftarrow} \mathcal{S}$ とし、s' を出力する。

実装したプログラムでは、3.4.2 節で説明するマルコフ連鎖のアルゴリズムを、大きい階数で利用することで、 nsgen と同等に近い機能を実現している。具体的には、形態素 7-gram のコーパスとその先頭の形態素が自立語となるものを要素とする配列を用意し、マルコフ連鎖を用いて文字列を生成する。階数 n が十分大きければ、マルコフ連鎖で生成される文字列は、文章源からそのまま切り出したものになる。7-gram を用いる理由は、コーパスのサイズと、出力される文字列の自然さのトレードオフを考慮[2]したためである。

3.4.2　ワードサラダ文識別問題

ワードサラダ文識別問題とは、人間が作成した自然文と、ワードサラダとを並べて証明者に提示し、ワードサラダを選ばせる問題である。

図 3.2 にアルゴリズムの概要、図 3.3 に自動作問結果例を示す。図 3.2 において、3.3 節、3.4.1 節で示した関数を使用しているが、読みやすさのため r_L, r_H, len_L, len_H の記述は省略している。同様に、3.4.3 節、3.4.4 節でも、こららの記述は省略する。

素材文章の取得に関して、図 3.2 1 行を説明する。文章源から、改行区切りの文章 m_j を要素とする素材文章の集合 $\{m_j\}_{j\in\mathbb{N}}$ を取得する。実装した作問プログラムでは、$\mathcal{M}$ に青空文庫を利用する。すなわち、青空文庫の著者一覧ファイル [11] からランダムに図書を選択し、該当する XHTML ファイルを解析して文章を取得する。読み込んだページにより、$|\{m_j\}_{j\in\mathbb{N}}|$ の値は変化する。

図 3.2 2–6 行は、n 階マルコフ連鎖用のコーパス（C_0）と Pre-Post-State 用のコーパス（C_1, PPS モデルと称す）の生成を示している。C_1 は、後述の離散型共起表現を C_0 に組み込む際に使用する。C_0, C_1 は、ある連想鍵に対して複数の連想値をもつ連想配列である。

図 3.2 4 行にある処理の詳細を示す。形態素解析により m_j の形態素配列 mor_j を取得する。mor_j の i 番目の形態素を $mor_j[i]$ とする。mor_j のサイズを +1 し、追加要素として、終端を表す特殊記号（終端記号と称す）を配列の最後に格納する。

C_0 の連想鍵は n-gram の形態素であり、連想値はそれから連鎖して文章に登場する形態素とその頻度情報である。C_0 の生成方法を、次に示す。

(1) $i \leftarrow 0$ とする。

(2) $i+n < |mor_j|$ ならば (3) に進む。そうでなければ、処理を終了する。

(3) 連想鍵を $key_0 \leftarrow (mor_j[i], \ldots, mor_j[i+n-1])$ とする。$mor_j[i+n]$ とその頻度情報に対して、次の判定をおこなう。

- key_0 が C_0 に未登録であれば、key_0 を連想鍵、形態素 $mor_j[i+n]$ と頻度情報 1 を連想値として登録する。

[2]本稿では、7-gram が最適だと主張はしない。この値は、一時的に採用したものである。

$\underline{(z, a) \leftarrow \mathcal{G}_{MCPT}(n, p, \mathcal{M})}$

n: マルコフ連鎖モデルの階数

p: 表示する文の数（選択肢数と一致）

$\mathcal{M}$: 文章源

(z, a): 問題文と解答のペア

Require: $p > 1$

1: $\{m_j\}_{j\in\mathbb{N}} \xleftarrow{\$} \mathcal{M}$
2: $C_0 \leftarrow \emptyset, C_1 \leftarrow \emptyset$
3: **for all** 文章 $m_j \in \{m_j\}_{j\in\mathbb{N}}$ **do**
4: 　m_j を形態素解析する。結果を、マルコフ連鎖用のコーパス C_0 と Pre-Post-State 用のコーパス C_1 に追加する。
5: 　m_j を構造解析し、係り受け関係がある 2 つの文節の組みを要素とする集合 $\mathcal{D}$ を生成する。
6: **end for**
7: **for all** 係り受けの文節の組み $d \in \mathcal{D}$ **do**
8: 　d と C_1 を用いて、C_0 に離散型共起表現を追加する。
9: **end for**
10: $x \xleftarrow{\$} (0, \ldots, p-1)$
11: C_0 から n 階マルコフ連鎖により、ワードサラダ s_x を生成する。
12: **for all** $i \in (0, \ldots, p-1) \backslash x$ **do**
13: 　$m \xleftarrow{\$} \{m_j\}_{j\in\mathbb{N}}$
14: 　$s_j \leftarrow \mathrm{nsgen}(m)$
15: **end for**
16: **for all** $i \in (0, \ldots, p-1)$ **do**
17: 　$z_i \leftarrow \mathrm{cogd}(s_i)$
18: **end for**
19: $z \leftarrow (z_0, \ldots, z_{a-1})$
20: $a \leftarrow x$

Fig. 3.2　Brief Algorithm of Markov-chain Phrase Test, $\mathcal{G}_{MCPT}$.

ワードサラダ文識別問題 1

(1) 七時過ぎに夕食を食べた。

(2) もし晴れたなら遊園地に困った!

(3) お金が足りず本が買えない。

(4) 益体のないことを考えていると、

正解: (2)

ワードサラダ文識別問題 2

(1) また、子牛は旅行とは速くならなければ

(2) バイクを下取りに出すときの気持ち

(3) 休日に一人で楽しんでもいいじゃないか

(4) 短距離が得意でもマラソンどうだろうか?

正解: (1)

ワードサラダ文識別問題 3

(1) 酒やジュースなど飲み物に入れるため

(2) 駅舎とホームは構内踏切で連絡している。

(3) 定休日と食べるの子どもは旅行と言えるの

(4) 治療の指標になる。この pH の測定は

正解: (3)

Fig. 3.3　Samples of Markov-chain Phrase Test without Consonant Gradation.

- key_0 が C_0 に登録済みだが、形態素 $mor_j[i+n]$ が連想値として未登録であれば、それと頻度情報 1 を連想値として追加する。
- key_0 と $mor_j[i+n]$ の組みが登録済みであれば、対応する頻度情報を +1 する。

(4) 連想鍵として登録された最初の形態素 $mor_j[i]$ が自立語の場合は、文頭可能属性としてマークを付ける。

(5) $i \leftarrow i+1$ とし、(2) に進む。

C_1 は、$mor_j[i-1], mor_j[i+1]$ の形態素を連想鍵とし、形態素 $mor_j[i]$ とその頻度情報を連想値とする。また、$mor_j[i-1], mor_j[i+1]$ それぞれの形態素に対する品詞・活用を連想記憶鍵とし、形態素 $mor_j[i]$ に対する品詞・活用とその頻度情報を連想記憶値とした構造も合わせ持つ。さらに、形態素と品詞・活用の対応を双方向で導出可能なマップを含む。C_1 については、C_0 の (1)–(3), (5) の処理を次の (1')–(3'), (5') で置き換え、(4) を省略して生成する。

(1') $i \leftarrow 1$ とする。

(2') $i+1 < |mor_j|$ ならば (3') に進む。そうでなければ、処理を終了する。

(3') 連想鍵と連想値の登録方法は、(3) と同様におこなう。ただし、形態素による鍵・値の構造に加え、その品詞・活用による鍵・値の構造を生成する。さらに、形態素と品詞・活用の対応を双方向で導出可能なマップを生成する。形態素から品詞・活用の情報は、MECAB や CABOCHA より取得する。

(5') $i \leftarrow i+1$ とし、(2') に進む。

次に、先行して 図 3.2. 11 行のワードサラダ生成方法について詳細を述べる。使用する表記について説明する。$val \overset{D}{\leftarrow} C(key)$ は、key に対する連想値として登録された要素を、その頻度分布 D に従い C からランダムに取得することを意味する。例えば、key に対して val_0 が頻度 70、val_1 が頻度 30 と登録されていた場合、val には 70% の確率で val_0、30%の確率で val_1 が代入される。

ワードサラダの生成方法は、次のようになる。

(1) 生成文字列長の下限の設定: 文字列長の下限を $len \overset{\$}{\leftarrow} \{len_L, \ldots, len_H\}$ とする。

(2) 文頭要素の取り出し: C_0 からランダムに文頭可能属性を持つ連想鍵 key_0 を取り出す。key_0 は n-gram の形態素であることに注意を要する。$val_0 \overset{D}{\leftarrow} C_0(key_0)$、$ary \leftarrow (key_0, val_0)$ とする。

(3) 終了判定: val_0 が終端記号であるか、ary に格納された総文字数が len 以上ならば処理を終了し、ary に格納された文字列を返す。そうでなければ、(4) に進む。

(4) 連鎖する形態素の取得: ary の最後から n 個の要素を連想記憶鍵　key_0 として取り出す。$val_0 \overset{D}{\leftarrow} C_0(key_0)$、$ary \leftarrow (ary, val_0)$ とし、(3) に進む。

本方式は、形態素情報を残しつつ再帰処理でワードサラダを生成するので、後述の離散型共起表現を組み込んだ場合でも、同様の処理がおこなえる。

最後に、図 3.2 5,7–9 行に示される、離散型共起表現への対応について述べる。例えば、文中で「もし」の後には「ならば」がよく出現する。しかし、2 つの語の間隔が長く空くこともある。長い間隔にも対処するには、階数 n を大きくして合成文を生成する必要がある。これは、ランダム性が弱くなるというトレードオフを生む。一方で、離散型共起表現への対処を放棄しようにも、それをワードサラダの発見の手がかりとするロボットの攻撃 [40] が既に提案されているため、この問題は無視できない。

この問題を解決するため、構造解析により、係り受け構造をもつ 2 つの文節の組みを要素とする集合 $\mathcal{D}$ を取得する。係り受け構造は、文節間距離が離れた場合でも発生するので、本稿では離散型共起表現が含まれていると考える。次に、C_0 への組み込みについて述べる。$d \in \mathcal{D}$ となるすべての要素に次の処理をおこなう。

(1) d を係り側の文節 seg_0 と受け側の seg_1 に分ける。これらは、形態素を要素とする配列で構成されているとする。$k_0 = |seg_0|, k_1 = |seg_1|$ とする。

(2) seg_0 を構成する最初の形態素 $seg_0[0]$、 最後の形態素を $seg_0[k_0 - 1]$ とする。seg_1 についても、同様に $seg_1[0], seg_1[k_1 - 1]$ とする。

(3) $key_1 = (seg_0[k_0 - 1], seg_1[0])$ を連想鍵とし、C_1 に照合する。

- 対応する連想値が存在すれば、$val \overset{D}{\leftarrow} C_1(key_1)$ とする。
- そうでなければ、$(seg_0[k_0 - 1], seg_1[0])$ に対する品詞・活用の組み合わせを PPS モデル内のマップから取得し、それを連想鍵 $\hat{key}_1$ として C_1 に照合する。
  - 対応する連想値が存在すれば、その品詞・活用 $\hat{val}$ を $\hat{val} \overset{D}{\leftarrow} C_1(\hat{key}_1)$ と取得する。PPS モデル内のマップに $\hat{val}$ を問い合わせ、対応する形態素を val に代入する。
  - そうでなければ、val に空文字を代入する。

(4) 形態素を要素とする配列を $ary = (seg_0, val, seg_1)$ のように生成する。

(5) $k = |ary|$ とする。C_0 に、連想鍵 $(ary[0], \ldots, ary[n - 1])$、連想値 $ary[n], \ldots, ary[k - 1]$ の組み合わせを登録する。$ary[0]$ が自立語ならば、文頭可能属性としてマークする。

(3) は、係り受け関係のある文節間に、PPS モデル C_1 に従い、形態素をランダムに挿入している。この処理は、単に 2 つの文節を接続し、ランダム性が弱くなることに対処している。(5) は、離散型共起表現を、あたかもひとつの形態素の組みのごとく扱い、C_0 に組み込んでいる。

3.4.3 機械翻訳文識別問題

機械翻訳文識別問題は、人間が作成した自然文と、機械翻訳文を並べて証明者に提示し、機械翻訳文を選ばせる問題である。機械翻訳文は、人間が作成した文に比べて、言い回しがぎこちない文となる。提案方式では、人間がこの違和感を識別できることを期待する。その文法や内容は、元の文が正

$\underline{(z, a) \leftarrow \mathcal{G}_{MTPT}(p, \mathcal{M}_{L_0}, \mathcal{M}_{L_1})}$

p: 表示する文の数（選択肢数と一致）

$\mathcal{M}_{L_0}$: 言語 L_0 （日本語）の文章源

$\mathcal{M}_{L_1}$: 言語 L_1 （英語）の文章源

(z, a): 問題文と解答のペア

Require: $p > 1$

1: $\mathcal{M}_{MT} \leftarrow \emptyset$
2: $\{m_j\}_{j\in\mathbb{N}} \xleftarrow{\$} \mathcal{M}_{L_1}$
3: **for all** 文章 $m_j \in \{m_j\}_{j\in\mathbb{N}}$ **do**
4: 　m_j を機械翻訳する。取得した文に nsgen を適用した結果を $\mathcal{M}_{MT}$ に追加する。
5: **end for**
6: $x \xleftarrow{\$} (0, \ldots, p-1)$
7: $s_x \xleftarrow{\$} \mathcal{M}_{MT}$
8: $\{m_j\}_{j\in\mathbb{N}} \xleftarrow{\$} \mathcal{M}_{L_0}$
9: **for all** $i \in (0, \ldots, p-1) \backslash x$ **do**
10: 　$m \xleftarrow{\$} \{m_j\}_{j\in\mathbb{N}}$
11: 　$s_j \leftarrow \mathrm{nsgen}(m)$
12: **end for**
13: **for all** $i \in (0, \ldots, p-1)$ **do**
14: 　$z_i \leftarrow \mathrm{cogd}(s_i)$
15: **end for**
16: $z \leftarrow (z_0, \ldots, z_{p-1})$
17: $a \leftarrow x$

Fig. 3.4　Brief Algorithm of Machine-translated Phrases Test, $\mathcal{G}_{MTPT}$.

機械翻訳文識別問題 1

(1) 経済が 2 パーセント成長すると予測します。

(2) ビールのボトルケースを買ったとき、

(3) はじめまして！よろしくお願いします！

(4) 少年と明日よく再び掛かっていること。

正解: (4)

機械翻訳文識別問題 2

(1) 飲めば飲むほどにやめられない味だ。

(2) 彼女は最も有名な家族の 1 つから来る。

(3) 私は毎日マラソンをする習慣がある。

(4) なんでもお気軽にご相談ください。

正解: (2)

機械翻訳文識別問題 3

(1) 交互の調子が悪い宇宙から何かを出すんだ。

(2) それから、正確に狙いをつけます。

(3) 私は彼らを追いかけた。すると、

(4) お腹がすいた、お昼になにを食べようか？

正解: (1)

Fig. 3.5　Samples of Machine-translated Phrases Test without Consonant Gradation.

常の文であり、かつ機械翻訳が原文に忠実である限り、大きな欠陥が出ない。自然文と機械翻訳文とを見分ける手がかりは、修辞の良し悪しの差だけになり、これはロボットには識別困難であると期待できる。

例として「Tell me, if you will, a little of your background.」という文を挙げ、人間と機械による翻訳の違いを示す。

(1) 人間による翻訳の例

- よろしければ、ご経歴を教えて下さい。

(2) 機械による翻訳の例

- 少し背景で、あなたがそうするならば、私に言ってください。
- あなたがそうするならば、私にあなたの背景の少しを話してください。
- する場合は、少し背景のことを、私に伝えてください。

機械翻訳文を使うテストの研究[41]として、利用者がテストを解く時に文章を作成させ素材文章を補給するものがある。この方式では、補給された言語 L_0 で書かれた文を機械翻訳を使って他言語 L_1 に一旦翻訳し、これを逆翻訳して元の言語 L_0 の文に戻し、違和感のある文章として利用するというアイデアである。

我々の提案する方式について、図 3.4 にアルゴリズムの概要を、図 3.5 に自動作問例を示す。本稿では、L_0 を日本語、L_1 を英語とする。提案方式では、インターネットから言語 L_1 の文章を直接取得し、機械翻訳により言語 L_0 の翻訳文を取得する。

言語 L_1 の素材文章とその機械翻訳文の集合 $\mathcal{M}_{MT}$ の取得に関して、図 3.4 1–5 行を説明する。翻訳前の素材文章は、改行区切りの文章の集合として扱う。実装した作問プログラムでは、英語版 Wikipedia[10] の「Random article」のリンク先を解析して文章を取得する。機械翻訳については、オンラインの機械翻訳を利用する。実装した作問プログラムでは、改行区切りの文章をフォームに入力し、POST 送信後の応答を解析し、翻訳文を取得する。

言語 L_0 の素材文章の取得に関して、図 3.4 8 行を説明する。日本語版 Wikipedia の「おまかせ表示」を用いる以外は、基本的には言語 L_1 と同じ処理をしている。

機械翻訳とその性能

機械翻訳として実際に使用した Excite 翻訳[3]や Weblio 翻訳[2]について述べる。これらの翻訳精度は、機械翻訳文識別問題の作問精度に影響を及ぼすが、オンラインサービスであるが故に、バージョン管理による再現実験には不向きである。

そこで、これらの翻訳性能について簡単に評価すると共に、今回の簡易評価で同程度の性能を持つオフライン動作可能なフリーの翻訳ソフト Yamato 英和 .NET Lite Ver.1.08[9] を紹介する。

翻訳性能の指標には、スムージング BLEU+1[29] を用い、原文とその正解となる和訳文のコーパスには、AAMT 機械翻訳文テストセット[1]を用いた。このコーパスでは、正解文が各原文につき 1 つしかないため、BLEU の評価としては、簡易的なものでしかないことに注意を要する。

結果のみを示すと、各指標値は、Excite 翻訳 0.319、Weblio 翻訳 0.311、Yamato 英和 .NET Lite Ver 1.08 が 0.314 であった。Mann-Whitney の U 検定にて、これらに有意性が無いことを確認した。

$\underline{(z, a) \leftarrow \mathcal{G}_{TDT}(p, q, \mathcal{K}_t, \mathcal{M})}$
p: 選択肢の数
q: 表示する文の数
$\mathcal{K}_t$: キーワードの集合
$\mathcal{M}$: 文章源
(z, a): 問題文と解答のペア
Require: $p > 1, q > 2, |\mathcal{K}_t| \geq p$
$key_t \overset{\$}{\leftarrow} \mathcal{K}_t$
key_t の類似語の集合 $\hat{\mathcal{K}}_t$ を取得する。
$\mathcal{K}_t \leftarrow \mathcal{K}_t \backslash key_t, \mathcal{K}_d \leftarrow \emptyset, \hat{\mathcal{K}}_d \leftarrow \emptyset$
while $|\mathcal{K}_d| < p - 1$ **do**
    $key_d \overset{\$}{\leftarrow} \mathcal{K}_t \backslash \mathcal{K}_d$
    $\mathcal{K}_d \leftarrow \{\mathcal{K}_d, key_d\}$
    key_d の類似語の集合を取得し、 $\hat{\mathcal{K}}_d$ に追加する。
end while
$\mathcal{S} \leftarrow \emptyset$
while $|\mathcal{S}| < q - 1$ **do**
    for all $\hat{key}_t \in \hat{\mathcal{K}}_t$ **do**
        $\mathcal{M}$ に対して、$\hat{key}_t$ を含み key_t と $\mathcal{K}_d$ の要素を含まない条件で検索する。取得した文章に nsgen を適用した結果を $\mathcal{S}$ に追加する。
    end for
end while
$x \overset{\$}{\leftarrow} (0, \ldots, q-1]$
for all $i \in (0, \ldots, q-1) \backslash x$ **do**
    $s_i \overset{\$}{\leftarrow} \mathcal{S}$
end for
$\hat{key}_d \overset{\$}{\leftarrow} \hat{\mathcal{K}}_d$
$\mathcal{M}$ に対して、$\hat{key}_d$ を含み key_t と $\mathcal{K}_d$ の要素を含まない条件で検索する。取得した文章に nsgen を適用した結果を s_x に代入する。
for all $i \in (0, \ldots, q-1)$ **do**
    $z_i \leftarrow \mathrm{cogd}(s_i)$
end for
$(z_0, \ldots, z_{q-1})$ に key_t と $\mathcal{K}_d$ を選択肢として加えたものを z に代入する。
$a \leftarrow key_t$

Fig. 3.6　Brief Algorithm of Topic Detection Test, $\mathcal{G}_{TDT}$.

共通話題識別問題 1

選択肢: (1) 政治、(2) 健康、(3) 経済、(4) 天気

- 国務をきちんとこなして欲しい。
- 参院態勢でみんな混乱　日本はどうなる？
- 資本主義社会では、経済理想をもって、
- 健康診断に費用がかかる。

正解: (1)

共通話題識別問題 2

選択肢: (1) 文学、(2) 社会、(3) 歴史、(4) 農業

- 読み物としてもおもしろい農学書
- 正史にその創作は含まれていない。
- 都市の変遷が伺える
- 経験がその人の年輪として刻まれている

正解: (3)

共通話題識別問題 3

選択肢: (1) 運動、(2) 探検、(3) 教育、(4) 金融

- 財務会計論が難しい …
- 売り手が投機筋しか見つからない
- 趣味にローンするのは反対
- 相手を探し、試合を開始しなさい。

正解: (4)

Fig. 3.7　Samples of Topic Detection Test without Consonant Gradation.

3.4.4 共通話題識別問題

共通話題識別問題とは、共通する話題の文脈に現れる文章を複数個、証明者に提示し、共通話題が何であるかを答えさせる問題である。

図 3.6 にアルゴリズムの概要を、図 3.7 にプログラムにより自動的に作問した結果例を示す。

共通話題（キーワード）の選択とその類似語の取得に関して、図 3.6 1–11 行を説明する。作問者は、知識非依存性要件を満たすように、キーワードの候補となる集合 $\mathcal{K}_t$ を入力する。作問アルゴリズムは、正解のキーワード key_t と偽のキーワードの集合 $\mathcal{K}_d$ を $\mathcal{K}_t$ から選択する。選択されたキーワードは、出題文の選択肢として使用される。それぞれのキーワードについて、シソーラスに問い合わせをおこない類似語の集合を取得する。

実装した作問プログラムでは、フォームにキーワードを入力した POST 通信を Weblio 辞書 [2] におこない、その応答を解析して $\hat{\mathcal{K}}_t, \hat{\mathcal{K}}_d$ を取得する。

次に、素材文章の取得に関して、図 3.6 12–17, 22–23 行を説明する。作問プログラムは、検索 API[4] を利用する。key_t や $key_d \in \mathcal{K}_d$ に含まれる語句は NOT 条件として検索し、類似語 $\hat{key}_t$ や $\hat{key}_d$ を含む文を取得する。よって、選択肢にあるキーワードが露骨に含まれる問題文の生成を抑止できる。

最後に、問題文の生成方法について説明する。図 3.6 18–21, 23 行では、1 文を偽の話題とし、残りの文を正解となる話題とし、証明者に提示する文章を生成している。偽の話題の文が少数混入することを証明者に告知して、多数を占める共通話題を答えさせる方式にすることで、ロボットによる解答をより困難にしている。

4　評価・考察・議論

本章では、3 章で示した提案方式を評価する。また、評価結果について考察すると共に、今後の課題となる点についての議論をおこなう。

4.1 節では、提案方式が人間に実際に解けるかどうかを、被験者による実験によって評価する。4.2 節は、提案方式に対する攻撃について分析をおこなう。4.3 節は、提案方式の問題新規性を評価する。4.4 節と 4.5 節では、それぞれ考察と議論をおこなう。

4.1　被験者らによる識別性要件の評価

4.1.1　予備実験: Google 音声型 CAPTCHA

予備実験として、Google のアカウント作成の際に要求される音声型 CAPTCHA の現状を調べた。本学に所属する 24 名が、被験者として参加した。視覚障害の程度は、全盲 7 人と弱視 17 名であり、全員が日本語を母国語とする。

被験者は、実際に Google のアカウント作成ページにアクセスし、問題に挑戦した。結果を表 4.1 に示す。このテストは、2013 年 10 月 25 日付近に改定され、それ以前に比べて難易度が若干緩和された。しかしながら、それでも 7% 程度の正答率に留まり、被験者の 79% は 10 回中 1 回も正答できない結果となった。

Table 4.1　Experiment for reCAPTCHA.

Version of reCAPTCHA	Number of Provers	Rate [%] of Provers who Answer Correctly at Least One Time	Rate [%] of Correct Answer
Before Oct. 25, 2013	5	0	0
Latest at Nov. 2013	24	21	7

4.1.2　実験環境

図 4.1 に、実験手順のシーケンス図を示す。

実装した作問プログラムにより、素材文章の収集と作問を事前におこない、被験者にはテキストファイル形式で提示する。作問は、テスト方式ごとに子音交替あり／なしに分けておこない、問題数は 10 問ずつとする。また、各問題の解答方式は、全て 4 択の択一選択方式とする。提案する CAPTCHA システムで人間を「ロボットではない」と判断するしきい値は、7 回とする。この基準は、総当り攻撃に対する耐性を考慮したものである。詳細は、4.2.1 節に示す。

4.1.3　ワードサラダ文識別テスト

3.4.2 節で述べたワードサラダ文識別問題を用いたテストが人間に解けるかどうかを評価した。本学に所属する 24 名が、被験者として参加した。視覚障害の程度は、全盲 7 人と弱視 17 名であり、全

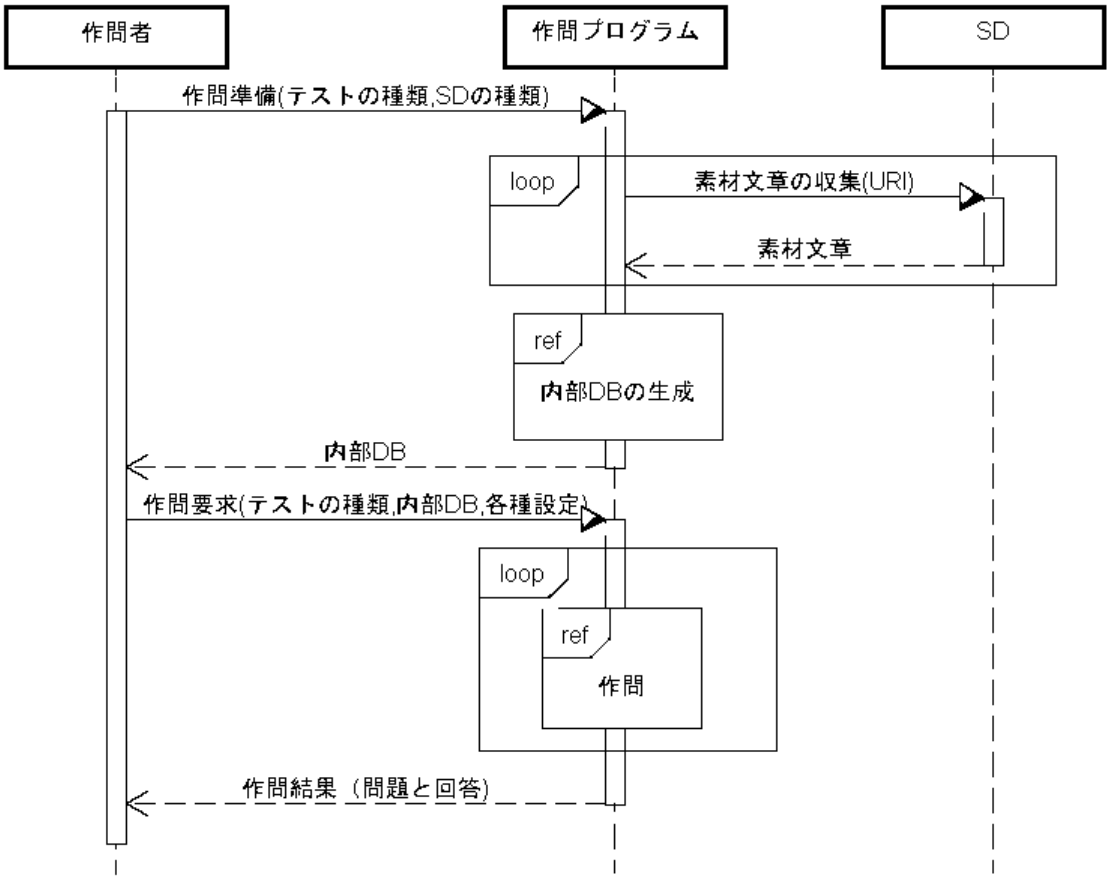


Fig. 4.1　Procedure of Subjective Experiment.

員が日本語を母国語とする。

$\mathcal{G}_{MCPT}$ の入力は、次のように設定した。実際に $\mathcal{M}$ から取得した素材文章は、1717 行、87260 文字、形態素 4361 種であった。

- $(n, p) = (1, 4)$
- $\mathcal{M}$ = “青空文庫”.

nsgen（3.4.1 節）の設定は、$(len_L, len_H) = (40, 80)$ とし、問題文 1 行あたりの文字数を 40–80 とした。cogd（3.3 節）の設定は、$(r_L, r_H) = (2, 5)$ とし、問題文 1 行あたり 2–5 箇所を子音交替の対象とした。

表 4.2 に結果の要旨を、図 4.2 に FAR (他人受入率: False Acceptance Rate) と FRR (本人拒否率: False Rejection Rate) の詳細を示す。表 4.2 の「Accuracy Rate」は、全問題に対する被験者全員の平均正答率を示している。「Rate of Ability as a Turing Test」は、10 問中 7 問以上正当できた被験者の割合を示している。なお、FAR に総当り攻撃を採用できる理由は、4.4.2 節に示している。

Table 4.2　Results of Markov-chain Phrases Test.

Sight	w/o Consonant Gradation		w/ Consonant Gradation	
	Accuracy Rate [%]	Rate [%] of Ability as a Turing Test	Accuracy Rate [%]	Rate [%] of Ability as a Turing Test
Totally Blind	89	86	73	**57**
Low Vision	78	82	49	**41**
All	85	83	60	**46**

4.1.4　機械翻訳文識別テスト

3.4.3 節で述べた機械翻訳文識別問題を用いたテストが人間に解けるかどうかを評価した。本学に所属する 12 名が、被験者として参加した。視覚障害の程度は、全盲 5 人と弱視 7 名であり、全員が日本語を母国語とする。

$\mathcal{G}_{MTPT}$ の入力は、次のように設定した。実際に取得した素材文章は、$\mathcal{M}_{L_0}$ の日本語文章が 48813 文字、1267 行、形態素 4019 種であり、$\mathcal{M}_{L_1}$ の英語単語が 1990 種であった。

- $p = 4$
- $\mathcal{M}_{L_0}$ = “日本語版 Wikipedia”.
- $\mathcal{M}_{L_1}$ = “英語版 Wikipedia”.

nsgen（3.4.1 節）の設定は、$(len_L, len_H) = (40, 80)$ とし、問題文 1 行あたりの文字数を 40–80 とした。cogd（3.3 節）の設定は、$(r_L, r_H) = (2, 5)$ とし、問題文 1 行あたり 2–5 箇所を子音交替の対象とした。

表 4.3 に結果の要旨を、図 4.3 に FAR と FRR の詳細を示す。

Table 4.3　Results of Machine-translated Phrases Test.

Sight	w/o Consonant Gradation		w Consonant Gradation	
	Accuracy Rate [%]	Rate [%] of Ability as a Turing Test	Accuracy Rate [%]	Rate [%] of Ability as a Turing Test
Totally Blind	28	33	28	**0**
Low Vision	30	33	30	**0**
All	55	33	28	**0**

4.1.5　共通話題識別テスト

3.4.4 節で述べた共通話題識別問題を用いたテストが人間に解けるかどうかを評価した。本学に所属する 24 名が、被験者として参加した。視覚障害の程度は、全盲 7 人と弱視 17 名であり、全員が日本語を母国語とする。

$\mathcal{G}_{TDT}$ の入力は、次のように設定した。実際に $\mathcal{M}$ から取得した素材文章は、1209 行、109042 文字、形態素 9726 種であった。

- $(p, q) = (4, 5)$
- $\mathcal{K}$ = {“スポーツ”, “天気”, “経済”, “食事”}
- $\mathcal{M}$ = “検索 API[4] で取得可能なインターネット上の文章”.

nsgen（3.4.1 節）の設定は、$(len_L, len_H) = (30, 60)$ とし、問題文 1 行あたりの文字数を 30–60 とした。cogd（3.3 節）の設定は、$(r_L, r_H) = (2, 5)$ とし、問題文 1 行あたり 2–5 箇所を子音交替の対象とした。

表 4.4 に結果の要旨を、図 4.4 に FAR と FRR の詳細を示す。

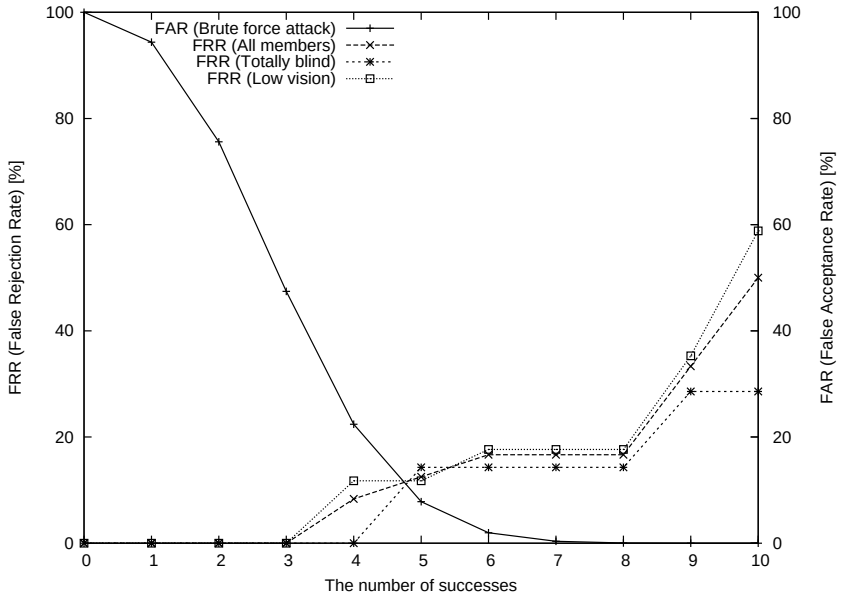
Table 4.4　Results of Topic Detection Test.

Sight	w/o Consonant Gradation		w/ Consonant Gradation	
	Accuracy Rate [%]	Rate [%] of Ability as a Turing Test	Accuracy Rate [%]	Rate [%] of Ability as a Turing Test
Totally Blind	71	86	81	**100**
Low Vision	67	71	60	**71**
All	69	75	70	**79**

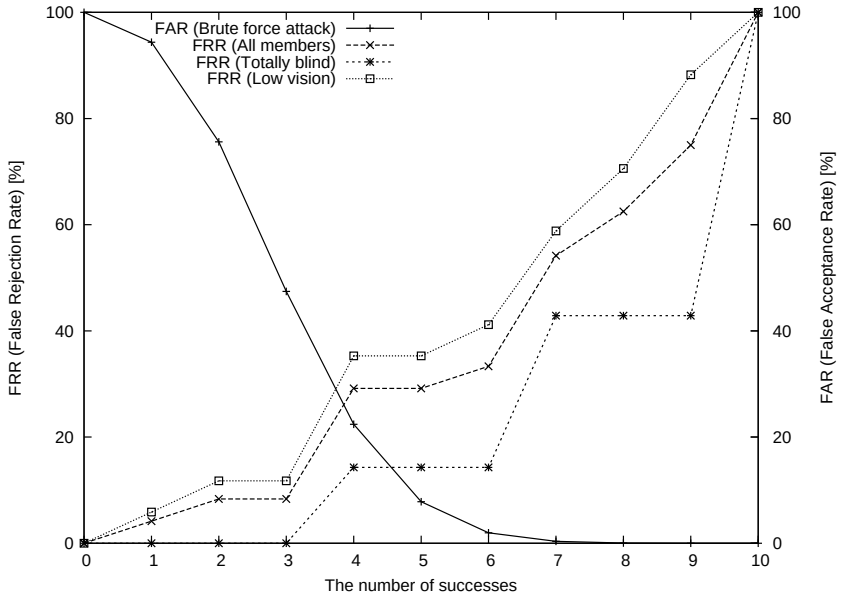
4.2　ロボットによる識別性要件の評価

4.2.1　総当り攻撃への耐性

4.1 節の設定で生成した各種問題について、ロボットによる攻撃が困難であることを示す。はじめに、総当たり攻撃への耐性を評価する。

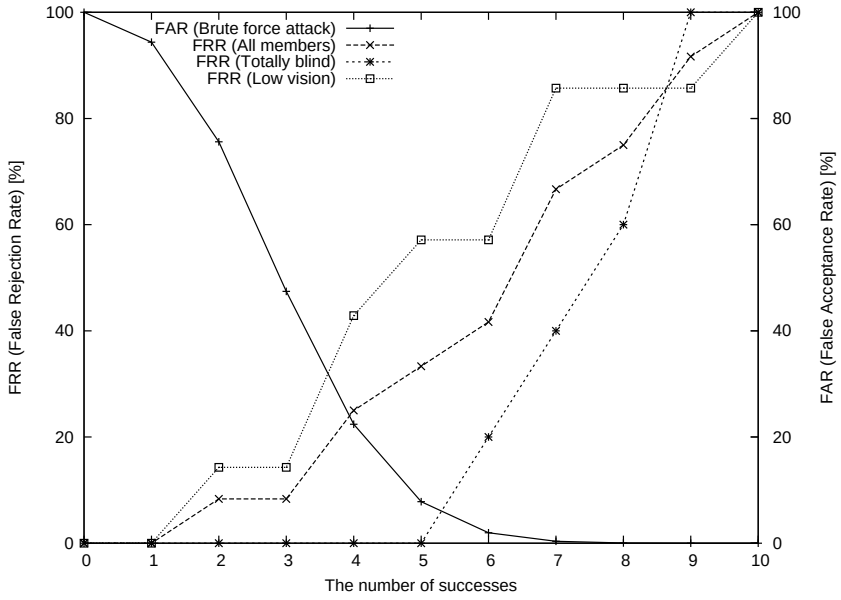


(a) Without Consonant Gradation.

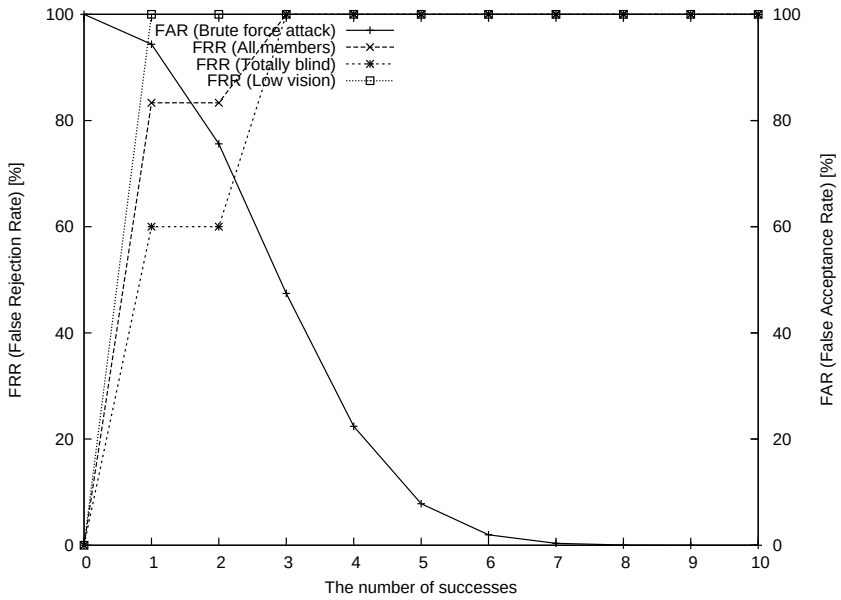


(b) With Consonant Gradation.

Fig. 4.2　FAR and FRR of Markov-chain Phrase Test.

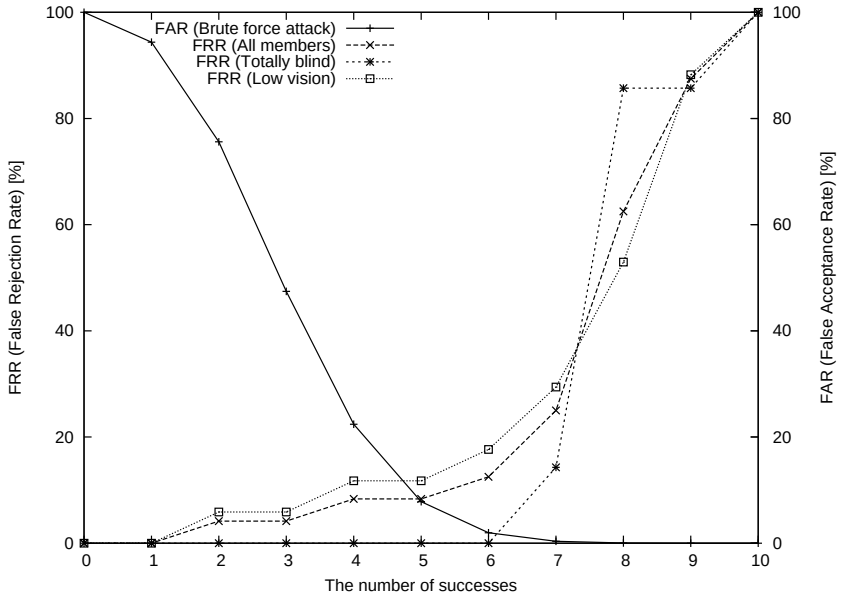


(a) Without Consonant Gradation.

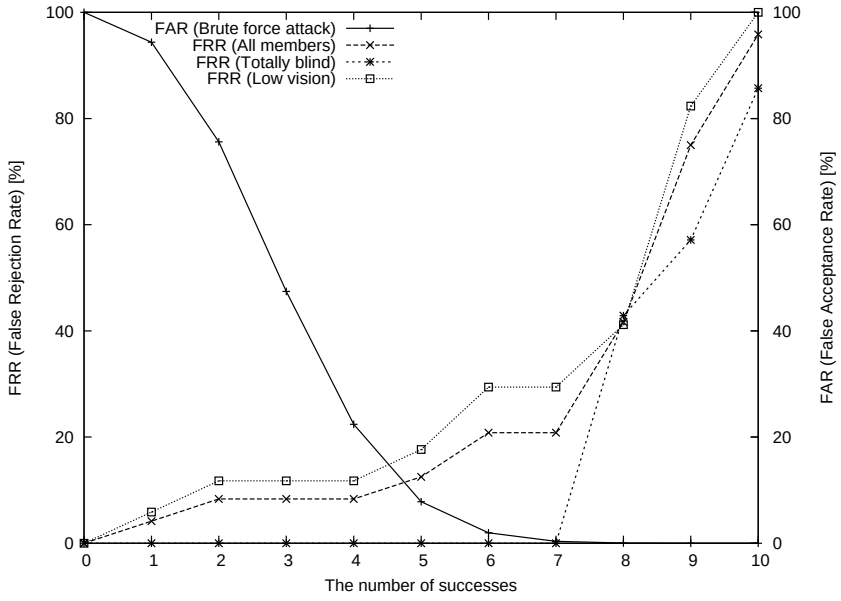


(b) With Consonant Gradation.

Fig. 4.3　FAR and FRR of Machine-translated Phase Test.



(a) Without Consonant Gradation.



(b) With Consonant Gradation.

Fig. 4.4　FAR and FRR of Topic Detection Test.

CAPTCHA システムが「ロボットでない」と証明者を判別する条件を、AI 問題 $\mathcal{Q}$ に n 回挑戦し k 回以上の正答とする。このようなテストを $CAPTCHA(\mathcal{Q}, n, k)$ と表す。アルゴリズム（ロボット）$\mathcal{P}^*$ が、$\mathcal{Q}$ 1 問に確率 P で解答できるとき、$\mathcal{P}^*$ が $CAPTCHA(\mathcal{Q}, n, k)$ の攻撃に成功する確率は、式 (4.1) で示される。

$$\mathrm{Succ}_{\mathcal{P}^*}^{CAPTCHA(\mathcal{Q},n,k)} := \sum_{i=k}^{n} \binom{n}{i} P^i (1-P)^{n-i} \tag{4.1}$$

Bursztein ら [14] によれば、$\mathcal{P}^*$ の攻撃成功確率が 1% 以上ある CAPTCHA は実用に耐えないとされているので、式 (4.2) がシステムとして要求される。

$$\mathrm{Succ}_{\mathcal{P}^*}^{CAPTCHA(\mathcal{Q},n,k)} < 0.01 \tag{4.2}$$

$\mathcal{Q}$ を 3 章で示した AI 問題とすれば、4.1 節で使用した設定では、解答の選択肢数は $p = 4$ なので、$P = 1/p = 1/4$ となる。このとき、式 (4.2) を満たす (n, k) は、$(10, 7)$ や $(4, 4)$ などがある。一方 4.1.2 節では、提案方式が証明者を「ロボットではない」と判別するには、10 問中 7 問の正答を要求するとした。したがって、4.1 節で使用した AI 問題は、総当たり攻撃に耐性がある。

4.2.2 子音交替適用前の文字列の特定困難性

子音交替後の文字列から、元の文字列を特定する攻撃を考える。元の文字列となる候補数は、式 (4.3) に従い、組み合わせ数を考えればよい。ただし、len は文字列長、r_L と r_H は、それぞれ子音交替適用回数の下限と上限である。

実験では、ワードサラダ文識別テストと機械翻訳文識別テストでは 40 文字、共通話題識別テストでは 30 文字を、文字列の最小長とした。また、子音交替は、文字列に対して 2–5 箇所をランダムに選び置換している。式 (4.3) より元の文字列の候補数は、$(len, r_H, r_L) = (40, 5, 2)$ のとき約 7.9 兆通り、$(len, r_H, r_L) = (30, 5, 2)$ のとき約 1.5 兆通りとなる。これは、180 万回/sec の処理能力を持つコンピュータを用いた場合でも、計算にそれぞれ 50、11 日を要する。

$$\text{”子音交替前の文字列の候補数”} = \sum_{i=r_L}^{r_H} \binom{len}{i} \tag{4.3}$$

4.2.3 検索による文章源特定の困難性

評価方法

検索攻撃耐性においては、次に示す評価法がる。

(1) 仮名の文章源の生成: 文章源内の文字列の漢字を仮名に開く。そうして取得した文章をウェブページとして公開し、既存の検索エンジンにそれらのページをインデックス化してもらう。

(2) 検索クエリの生成: (1) を文章源とし、3 章に示したアルゴリズムに従い、問題文を生成する。これらの子音交替を適用した文字列を検索語として、既存の検索エンジンに問い合わせる。

(3) 評価: 「あいまい検索」などの検索語の補正機能により、各検索語に相当する文章源が特定されたかを確認する。次の評価指標を用いて、複数サンプルに対して評価する。

+1: 第 1–10 候補に選出された。

+0: 選出されない、または第 11 候補以降に選出された。

この方法は、文章源の分量が多いと実現困難である。さらに、(1) のように公開されたページに対して高いページランクがつくとは考えにくく、(3) の評価結果が甘くなる恐れがある。

代替案 1: PPS モデルに従った文字置換に対する評価方法

前述の評価方法では、子音交替の際に漢字を仮名に開くため、(1) の作業が必要になり問題となる。よって、子音交替を適用しない、異なる文字置換アルゴリズムを用いた評価をおこなう。これは、提案方式の安全性を直接評価するものではないが、文字置換と置換箇所数において、似た結果を返すと期待できる。このアルゴリズムは、漢字や仮名を問わず、次のように文字を置換する。

(1) 素材文章の切り出し: 3.4.1 節で示した方式に従い、文章源から素材文章を切り出し 100 個の文字列を用意する。

(2) 文字置換: 文字列を s を、1 文字を要素とする配列構造とする。$s[i]$ を i 番目の要素とする。

子音交替の代わりに、3.4.2 節で述べた PPS モデルを利用した文字置換を、(1) で用意した全ての文字列に対して次のように処理する。

(2-1) $r \xleftarrow{\$} \{r_L, \ldots, r_H\}$ とし、置換回数を決める。4.1 節の設定に従い、$(r_L, r_H) = (2, 5)$ とする。

(2-2) $\mathcal{Z} \leftarrow \{0, \ldots, |s| - 1\}$ とする。$|s| < r$ であれば、$r \leftarrow |s| - 1$ とする。

(2-3) $r > 0$ ならば (2-4) に進む。そうでなければ、処理を終了する。

(2-4) $i \xleftarrow{\$} \mathcal{Z}$ とする。

(2-5) 置換対象の文字を $s[i]$ とする。$s[i-1], s[i+1]$ を連想鍵とし、PPS モデルから連想値 val を取得する。$s[i] \leftarrow val$ と置換する。

(2-6) $r \leftarrow r - 1$ として、(2-2) に進む。

既存の検索エンジンを用いて、(2) で生成した文字列を検索し、前述の指標に従い評価する。結果を表 4.5 に示す。PPS モデルに従った 2–5 箇所の文字置換は、検索エンジンによる文章源の特定を 30–40% まで困難にしている。

代替案 2: 仮名文字列を漢字に再変換して評価する方法

攻撃コストを抑えた現実的な攻撃は、子音交替処理により仮名のみとなった文字列を、漢字仮名交じりの文字列に再変換する方法である。

正しい漢字変換をするには、精度の高い形態素解析が必要になる。そこでまず、子音交替による形態素解析の妨害効果を調べる。

結果を図 4.5 に示す。横軸は、子音交替適用数の最大値 r_H の値を示す。なお最小値 r_L の値は、$r_H = 1$ であれば $r_L = 1$、そうでなければ $r_L = 2$ である。縦軸には、2.5 節で示したスムージング BLEU+1 の値を、子音交替前後の文字列の形態素ごとの類似度を表す指標として利用した。

Table 4.5 Rate [%] of Finding Sources owning to the Replacement Number r.

Search Engine	$r = 1$	$r = 2$	$r = 3$	$r = 4$	$r = 5$
Bing	19	18	20	15	12
Yahoo! Japan	47	41	40	28	30
Google	46	34	–†	–†	–†

†: We could not get correct results since the search engine refused our queries.

凡例「CG」のグラフは、子音交替後の文字列を、子音交替前の文字列を仮名に開いたものと比較した結果である。この場合、形態素解析結果の類似度は 10% 以下となり、元の情報はほとんど失われている。凡例「CG+SR」のグラフは、子音交替結果に音声認識を施し文字列の修復と漢字化をしたものと、子音交替前の文字列を比較した結果である。BLEU 値は、漢字を仮名に開いてから計算している。この処理は、音声認識での修復効果を用いて、「誤植」や「聞き間違い」を故意に発生させる子音交替処理を攻撃する場合を想定している。

音声認識による修復は、次のようにしておこなった。

(1) 素材文章の切り出し: 3.4.1 節で示した方式に従い、文章源から素材文章を切り出し 100 個の文字列を用意する。

(2) 音声化: Microsoft Speech Platform 11 (MSSP 11) により (1) で生成した文字列を読み上げ、48kHz、16bit、 ステレオの MP3 ファイル[1]に変換する。

(3) 音声認識: トレーニングを実施した Dragon Speech 11 を用いて、(2) で生成した MP3 を音声認識し、文字列として出力する。この作業は、PC 内部の音声ファイルを直接認識させた。よって、スピーカを通すことによる認識精度の低下は発生しない。

形態素解析結果の類似度は 20–40% になる。これは、凡例「CG」の場合に比べて、BLEU 値で 10–30% の修復がおこなわれている。4.1 節の設定の範囲で考えれば、子音交替は 20–30% まで類似度を低減していることがわかる。なお、グラフにおける子音交替数なしの BLEU 値の意味は、仮名に開くことによる形態素解析の妨害効果を示している。

次に、音声認識により修復された文字列を検索し、どの程度文章源を特定できるかを調べる。結果を図 4.6 に示す。横軸は、子音交替適用数の最大値 r_H の値を示す。なお最小値 r_L の値は、$r_H = 1$ であれば $r_L = 1$、そうでなければ $r_L = 2$ である。縦軸は、検索結果のスコアで、高いほど文章源の特定がされにくいことを示す。スコアの指標は、次のように設定した。

-1: 第 1 候補に選出された。

+0: 第 2–10 候補に選出された。

+1: 選出されない、または第 11 候補以降に選出された。

[1]作業の自動化に使用した Text To Wav[7] での最高音質である。

評価サンプル数は、各条件につき 100 個である。凡例は、子音交替の対象とした文字列が、自然文とワードサラダのどちらかを示している。

自然文に対して子音交替による置換を 2–4 箇所以上おこなうと、文章源の特定率は 2% 以下となる。この結果は、$n = 1$ 階マルコフ連鎖に基づき生成したワードサラダ文に対して子音交替をおこなったものと比較し、有意水準 5 % での Mann-Whitney の U 検定で有意差がない。ワードサラダと自然文の検索結果に有意差がなければ、攻撃者はそれを手がかりにすることができない。

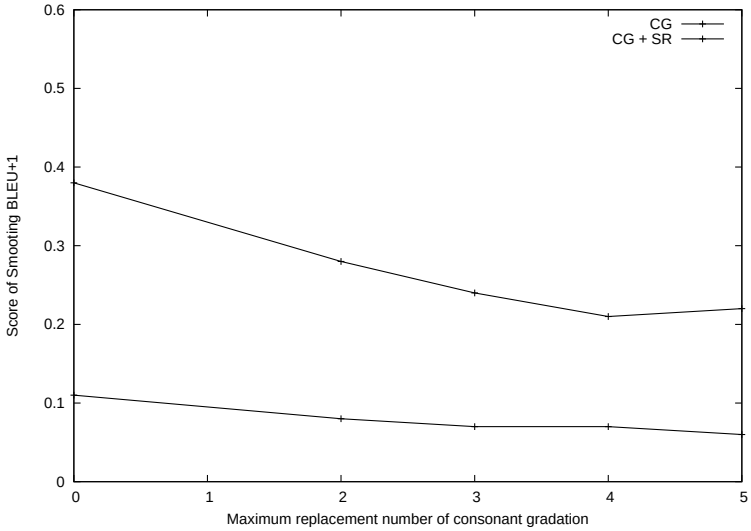


Fig. 4.5　Smoothing BLEU+1 Score of Morphological Analysis.

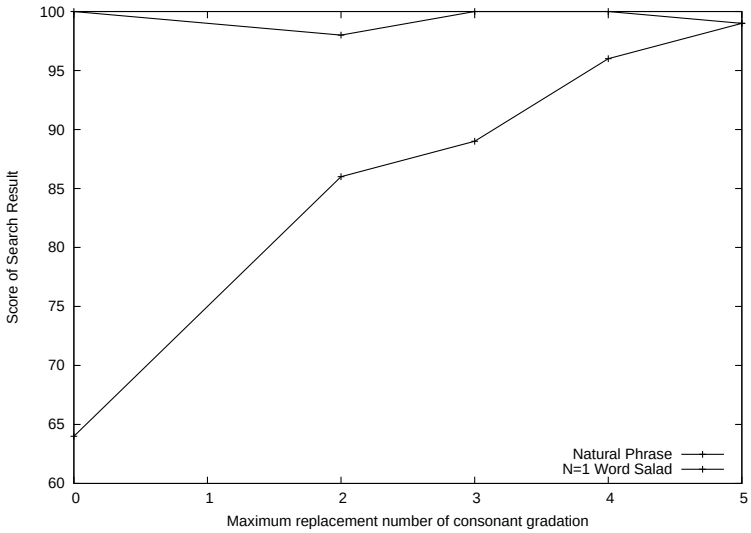


Fig. 4.6　Failure Score of Fiding Sources.

4.3　問題文新規性の評価

本研究では、各テスト方式ごとに 1 万回の作問を実施し、新規な文を生成する確率を、問題文新規性要件の評価指標として用いる。新規な文とは、過去に作問された問題文中に含まれないものを意味する。

結果は、ワードサラダ文識別テストが 99.94%、機械翻訳文識別テストが 99.59%、共通話題識別テストが 99.65 % であった。いずれの方式においても、99% 以上の新規文生成能力を備えている。

4.4　考察

4.4.1　各テスト方式間の比較

子音交替の影響

2.6 節に示した検定方法を使用する。この検定方式は、4.1 節の実験結果から、子音交替適用前（実験条件 1）と適用後（実験条件 2）の有意性を判定するために使用する。ワードサラダ文識別テストと共通話題識別テストの被験者数は 24 人なので、 $X = 10$ となる。機械翻訳文識別テストの被験者数は 12 人なので、$X = 8$ となる。実験結果から $|D|$ の値を計算すると、ワードサラダ文識別テストは $|D| = 21$、機械翻訳文識別テストは $|D| = 12$、共通話題識別テストは $|D| = 5$、になる。したがって、共通話題識別テストは子音交替のあり／なしによる有意差はないが、ワードサラダ文識別テストや機械翻訳文識別テストでは有意差が認められた。

この違いの理由について分析する。子音交替を自然な文に適用すると、文字置換がなされるため、不自然さが増す。ワードサラダ文識別テストや機械翻訳文識別テストは、文章の自然さを見極める問題なので、自然文としたものの不自然さが増加すると解答が難しくなると考えられる。一方、共通話題識別テストでは、文の大意さえ読み取れればよい。多少であれば、文が不自然になったところで文意の解釈は困難にはならないので、子音交替の影響が少ないと考えられる。また、この方式では、複数文から大意を占める話題を選択するので、仮に 1–2 文の文意が解釈できなくとも、他の文の解釈により解けるという特徴もある。

人間とロボットの判別能力

まず、被験者の解答パタンを、ランダム選択によるものと比較する。被験者らの解答は、有意水準 5%での Mann-Whitney の U 検定において、ランダムな解答との有意差が認められた。よって、今回の結果は、被験者らが適当に解答したものではなく、実験結果として評価可能である。

提案手法の人間とロボットの判別能力は、共通話題識別テストで 79%、ワードサラダ文識別テストで 46%であり、現状の Google 音声型 CAPTCHA に比べて、高確率で人間を「ロボットではない」と判別できている。この性能差は、有意水準 5%での Mann-Whitney の U 検定において有意であることから、提案方式の効果が確認できた。共通話題識別テストとワードサラダ文識別テストの間の性能差は、子音交替による影響が考えられる。

一方、機械翻訳文識別テストは、現状 CAPTCHA として機能しない結果となった。子音交替なしの状態でも正答率が悪いが、子音交替による悪影響で極端に低い正答率となった。対策としては、精度を意図的に落とした機械翻訳を使用したり、より機械翻訳が難しいと期待できる口語的文章の利用による改善が必要である。

提案方式の識別能力は、全盲者と弱視者に対する統計的な有意性はみられなかった。しかしながら、問題ごとの正解／不正解の傾向を調べると、Mann-Whitney の U 検定において有意性がみられた。このことは、被験者を増やすことで、全盲者と弱視者の間に、有意性が出る可能性を示唆している。実際、表 4.2 や表 4.4 からは、提案方式が全盲者に対してより高い識別能力を持つことがわかる。

問題新規性

いずれの方式も 99% 以上の新規文生成能力を備えるが、ワードサラダ文識別テストが最も優れていた。また、機械翻訳文識別テストと共通話題識別テストは、同程度の新規文生成能力であった。

機械翻訳文識別テストについては、「翻訳抜け」という問題も見受けられた。今回採用した Wikipedia では、そのサイトの性質から固有名詞が多く含まれており、翻訳されずにアルファベット記述が残る場合が多々見受けられた。評価プログラムでは、このようなテストに適さない文を生成した場合、自動的に作問をやり直す。これは、処理時間が増大するだけでなく、実質的に使用できる文章源が少ないことを示している。機械翻訳文識別テストの翻訳対象となる文章源の選定には、このような点にも注意を要する。

4.4.2　ロボットによる攻撃

子音交替適用前の文字列の特定

4.2.2 節では、元の文字列の特定が、現在のコンピュータでは困難であることを示した。実際の攻撃では、さらに文字列を文章源から検索し、問題のヒントを得る必要がある。しかし、 攻撃者は元の文字列を知らないので、復元した文字列が正解であることの確認手段が難しい。例えば、「駅についた」という文字列に子音交替を適用し、「めきについた」を得たとしよう。元の文字列の候補である「せきについた」、「てきについた」は正解ではないが、意味は通じてしまう。これらを検索すれば、それぞれの文章源が特定され、不正解であるとは言い切れない。実際に、元の文字列が「席についた」、「敵についた」（敵方についたの意味）であれば、これらは正解となる。このように、正確に元の文字列を復元し、その文章源を特定することは、困難な場合がある。

攻撃コストに制限を設けない検索攻撃による文章源の特定

漢字を仮名に書き換えた文章源のデータベースを持つ場合[2]を検討する。4.2.3 節に示した PPS モデルによる評価から、 検索攻撃による文章源の特定は 30–40 % 成功する。

著者は、この結果はほぼ最悪の場合を示しており、実際はより特定されにくいと考えている。まず、子音交替で得られる仮名文は、漢字仮名混じり文に比べ使用される文字種が少ないため、あいまいで特徴の少ない文になり検索による特定がしにくい。

形態素解析も困難になる。教師ありの形態素解析を用いるためには、仮名基準の辞書が必要になるが、これは整備されていない。一方、教師なしの形態素解析は一般には時間がかかるため、CAPTCHA への攻撃には使用できない。仮に形態素解析ができたとしても、子音交替による形態素解析妨害効果により、正しい結果を得るのが難しい。図 4.7 に、予め仮名に開いた文字列と、それに子音交替を適用して得た文字列を、スムージング BLEU+1 で評価した結果を示す。凡例の意味は、図 4.5 と同じである。結果は、音声認識による修復を入れても、10–20 % 程度の情報を喪失している。正しい形態素

[2]繰り返しになるが、仮名文字列のデータベースの準備は、攻撃コストが非常に高い。さらに、新規文章が湧出する Twitter などを文章源に用いれば、 攻撃者は常にデータベースの更新が必要となる。

解析ができなければ、検索語の修復などの高度な検索も困難になる。

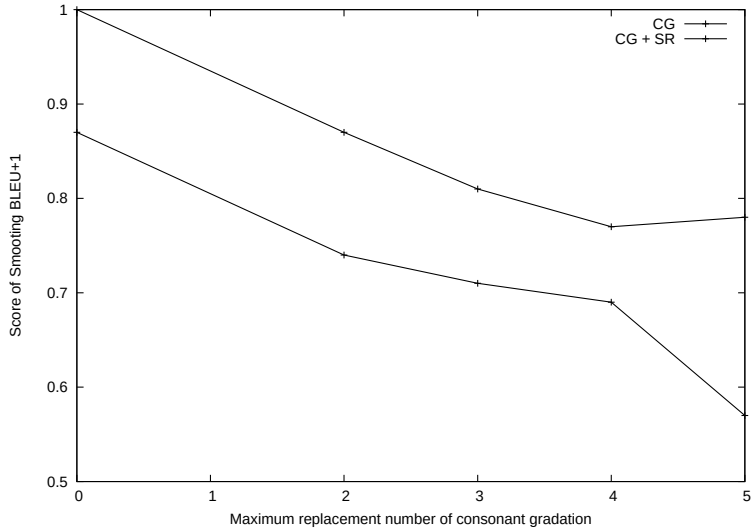


Fig. 4.7　Smoothing BLEU+1 Score Compared Sources Consist of Kana-strings with Strings after Consonant Gradation.

現実的な検索攻撃による文章源の特定

子音交替適用後の仮名文字列を漢字に再変換して攻撃する場合を検討する。図 4.6 から、文章源の特定率は 2% 以下にに留まる。4.1.2 節より、問題文は最大 5 行の文字列が提示される。これらは独立して生成されるため、どれか 1 つでも文章源が特定される確率は $2 \times 5 = 10[\%]$ となる。一方、選択肢数は 4 なので、1 問あたりのランダムな正答率は 25% である。したがって、検索攻撃はランダム選択攻撃よりも成功率が低いので、安全性におけるボトルネックにはならない。

また、元の文字列がワードサラダ／自然文のいずれにおいても検索結果に有意性が無いので、提案方式は、検索結果の有意性を用いた攻撃にも耐性を持つ。

KL ダイバージェンスを用いた共通話題識別テストへの攻撃

子音交替による形態素解析の妨害は、KL ダイバージェンスを用いた攻撃にも効果がある。なぜなら、正しい単語（形態素）を得られなければ、KL ダイバージェンスの値が信頼出来ないからである。

4.4.3　CAPTCHA システムのパラメータ検討

本章のまとめとして、2.7 節で定義した CAPTCHA システムのパラメータを考察する。

提案方式の安全性に影響を与えるパラメータは、子音交替の対象となる文字列長 *len*、子音交替適用回数の範囲 (r_L, r_H)、問題の提示回数 n、「ロボットでない」と判別するしきい値 k がある。セキュリティパラメータを、$\kappa := (len, r_L, r_H, n, k)$ とする。

$\epsilon_{\mathcal{P}}(\kappa)$ の値は、実験で得られた値を採用すると、ワードサラダ識別テストについては $\epsilon_{\mathcal{P}}^{\mathcal{G}_{MCPT}}(\kappa_{MCPT}) = 0.46$、共通話題識別テストについては $\epsilon_{\mathcal{P}}^{\mathcal{G}_{TDT}}(\kappa_{TDT}) = 0.79$ となる。ただし、$\kappa_{MCPT} = (40, 2, 5, 10, 7)$、$\kappa_{TDT} = (30, 2, 5, 10, 7)$ である。なお、機械翻訳文識別テストは、今回の結果では CAPTCHA としては使用できないので、パラメータの検討から除外した。

4.2 節の実験と 4.4.2 節の検討により、ロボットが検索から解答のヒントを得るのは困難である。よって、4.2.1 節に示した総当り攻撃への耐性を基準として、$\hat{\epsilon}_{\mathcal{P}^*}(\kappa) = 0.01$ とする。

$\hat{\epsilon}_{\mathcal{P}}$ については、4.4.2 節の検討で示したロボットの演算時間に満たない値にすればよい。利用者の可用性を損ねない程度に、1 問あたり数分にしておけばよいだろう。

4.5　議論

4.5.1　可用性

問題文の文字数

証明者 $\mathcal{P}$ に提示する 1 行あたりの表示文字数を $|\ell_{0,i}|$、 表示行数を q とすれば、1 回のテストで回答者が読まねばならない文字数は、式 (4.4) のようになる。$\ell(q)$ は小さいほど可用性は高いが、小さすぎると、識別性要件を満たす十分な情報が含まれない恐れがある。

$$\ell(q) = \sum_{i=0}^{q-1} |\ell_{0,i}| \tag{4.4}$$

機材の都合から $|\ell_{0,i}|$ の上限の目安は決まる。例えば、 点字ディスプレイは 1 行あたりに 40–80 文字を表示可能な製品が多い。したがって、$|\ell_{0,i}| \leq 80$ を制約[3]とするのが望ましい。

また、$|\ell_{0,i}|$ の下限を決定すべきであるが、人間が十分に高い確率で AI 問題を解くために必要な情報量の下限は、テスト方式や文章源の種類によって異なると推測される。本稿の実験では、著者が作問プログラムの実装中に得た経験を元に値を決定したが、今後はこの観点に主眼をおいた実験と解析が必要になる。

解答方式

提案方式では、4.1.2 節に示したように、解答方式として択一選択肢方式を採用している。これを変更することで、安全性を保ちつつ可用性を向上できる可能性がある。安全性の評価基準として、 式 (4.1) に示した総当り攻撃の成功確率を用いる。

選択肢数を p とすれば、択一選択方式における問題文 1 問あたりの総当たり攻撃成功確率は $P = 1/p$ となる。式 (4.2) を満たす最小の n は、$p = 4$ であれば $(n, k) = (4, 4)$ となる。逆に $n = 1$ とするためには、$p > 100$ が必要になる。

p 個の候補から固定数 j を解答する方式を採用すれば、$P = 1/{}_pC_j$ となる。式 (4.1) から、この解答方式は $(p, j) = (9, 4)$ の場合において、式 (4.2) を $(n, k) = (1, 1)$ で満たす。

p 個の候補から $j \in \{0, \ldots, p\}$ 個を解答する方式を採用すれば、$P = \sum_{i=0}^{p} {}_pC_i$ となる。式 (4.1) から、この解答方式は $p = 7$ の場合において、式 (4.2) を $(n, k) = (1, 1)$ で満たす。

ワードサラダ文識別テストや機械翻訳文識別テストでは、$p = q$ となる。p に対する P の値が小さいほど n を小さくできるので、式 (4.4) より文章量を削減することができる。

一方で、複数解答を許容する方式は、消去法や相対比較による解答ができなくなる。よって、人間による正答率も低下すると考えられる。このトレードオフについても、今後の評価実験を通して、最

[3]もちろん行送りはできるが、ワードサラダ文識別テストのように、問題文がそのまま選択肢になっている場合を考えると、画面と点字情報が一致している方が望ましい。

適なものを選択する必要がある。

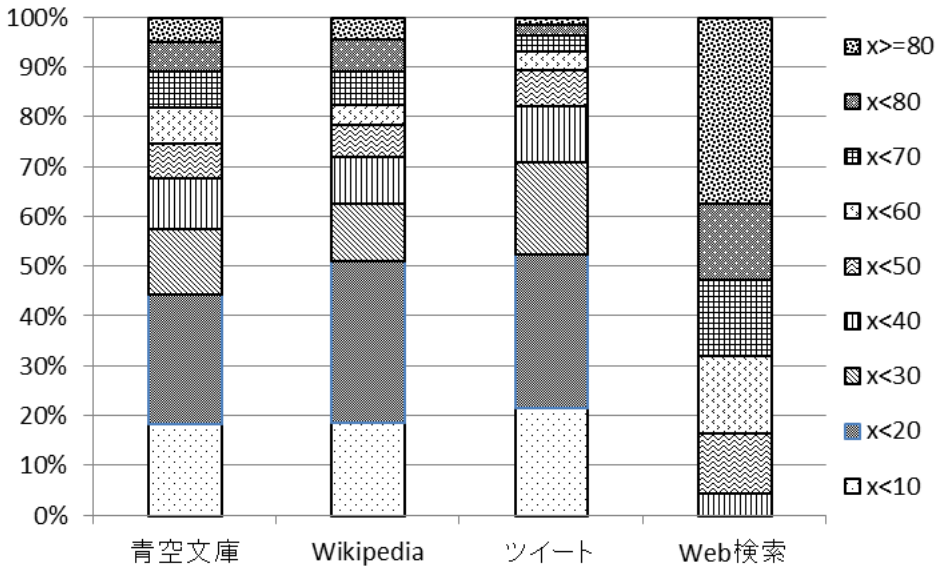


Fig. 4.8　The Length of String in Each Source Document.

4.5.2　文章源の違いによる問題文新規性への影響

4.5.1 節において、問題文 1 行あたりの文字数の重要性を示した。しかし、文章源から該当する長さの文字列を素材文章として収集できなければ、そもそも文字列長の制御ができない。また、文字列長の制限により、使用可能な文章源の分量に厳しい制限がついた場合、問題文新規性への影響も懸念される。

図 4.8 に、文章源ごとの段落（改行で分割した文章）に含まれる文字数のデータを示す。図の凡例は、$x < 10$ は 0–9、$x < 20$ は 10–19 のような範囲を表す。Wikipedia と Web 検索に関しては、4.1 節で利用したものの分布である。青空文庫については、4.1 節で利用したものに加え、さらにランダムに収集した 4 つの文章を加えたものである。ツイートに関しては、2013 年 5 月 15 日 19 時から 8 時間ほど収集した 13713 件のデータである。

例えばワードサラダ文識別テストでは、文章源に青空文庫を選択し、$|\ell_{0,i}|$ の下限を 40 にした。図 4.8 からすると、大雑把に見て少なくとも文章源の 40% を素材文章として 利用できる。もし、文章源を Wikipedia や ツイートに変更した場合、問題文新規性を維持するには、文章源からより多くの素材文章を収集しておく必要があるだろう。このように、$|\ell_{0,i}|$ を決める際は、文章源の選択と合わせて考える必要がある。

5　結論

本稿では、セキュリティ技術の特定知覚への依存の問題を取り上げ、代表例である CAPTCHA について、そのバリアフリー化についての研究に取り組んだ。

本研究では、バリアフリーな CAPTCHA を構成するための要件を論じ、その解決例を提示した。提案方式では、文意文脈解釈問題を用いることで知覚依存を解消する。問題新規性は、時々刻々作り出されるネット上の文章を作問に利用することで解決した。また、ネット上の公開文章を安全に使用するための問題点を指摘し、その対処法として子音交替による方法を示した。子音交替の適用結果は、「誤植」や「聞き間違い」と似たものとなり、人間は日常的な経験によりそれを修復できる。

さらに本研究では、視覚に障害のある被験者による実験をおこない、その効果を確認した。ワードサラダ文識別テストや共通話題識別テストについては、既存の Google 音声型 CAPTCHA に比べ、高い確率で人間を「ロボットではない」と判別できた。また、総当り攻撃、子音交替前の文字列の復元攻撃、そして検索攻撃に関する評価と検討をおこない、現在のロボット性能に対して、提案方式が安全であることを示した。最後に、今後の課題として、可用性、並びにに文章源と問題文新規性の関係について議論した。

謝辞

本研究は、著者が筑波技術大学大学院技術科学研究科在学中に、同大学保健科学部情報システム学科の岡本健准教授、および佐々木信之教授のもとでおこないました。本研究を遂行するにあたり、終始御指導くださり、心より感謝いたします。また、様々な指摘、助言を下さいました産業総合研究所セキュアシステム研究部門の中田亨氏、および明治大学大学院先端数理研究科の菊池浩明教授に心より感謝いたします。最後に、研究に関わりました関係者各位と家族の支援に深く感謝いたします。

参考文献

[1] AAMT 機械翻訳文テストセット, http://corpus.aamt.info/corpus/ja/corpus.html.

[2] Excite 翻訳, http://translate.weblio.jp/.

[3] Excite 翻訳, http://www.excite.co.jp/world/.

[4] GAPI.net 0.5.0.1, http://gapidotnet.codeplex.com/.

[5] The Official CAPTCHA Site, http://www.captcha.net/.

[6] Project Stiltwalke, http://www.dc949.org/projects/stiltwalker/.

[7] Text to Wave 3.083, http://noah.ninja-web.net/soft/index.html.

[8] Web Content Accessibility Guidelines (wcag) 2.0, http://www.w3.org/tr/wcag20/.

[9] Yamato 英和 .net lite ver.1.08, http://www.vector.co.jp/soft/win95/edu/se364617.html.

[10] 英語版 Wikipedia, http://en.wikipedia.org/wiki/main_page.

[11] 青空文庫の著者一覧ファイル, http://www.aozora.gr.jp/index_pages/list_person_all.zip.

[12] Jeffrey P. Bigham and Anna C. Cavender. Evaluating Existing Audio CAPTCHAs and an Interface Optimized for Non-visual Use. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2010), pages 1829–1838. ACM, 2009.

[13] Elie Bursztein, Steven Bethard, Celine Fabry, John C. Mitchell, and Dan Jurafsky. How Good Are Humans at Solving CAPTCHAs? A Large Scale Evaluation. Proceedings of the 2010 IEEE Symposium on Security and Privacy (SP 2010), pages 399–413. IEEE Computer Society, 2010.

[14] Elie Bursztein, Matthieu Martin, and John Mitchell. Text-based CAPTCHA Strengths and Weaknesses. Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS 2011), pages 125–138. ACM, 2011.

[15] Philip K. Dick. Do Androids Dream of Electric Sheep? Ballantine Books, 1968.

[16] George Doddington. Automatic Evaluation of Machine Translation Quality Using N-gram Co-occurrence Statistics. Proceedings of the Second International Conference on Human Language Technology Research (HLT 2002), pages 138–145. Morgan Kaufmann Publishers Inc., 2002.

[17] Jeremy Elson, John R. Douceur, Jon Howell, and Jared Saul. Asirra: A CAPTCHA that Exploits Interest-aligned Manual Image Categorization. Proceedings of the 14th ACM conference on Computer and communications security (CCS 2007), pages 366–374. ACM, 2007.

[18] Frank Clay Fisk, Sri Ramanathan, Matthew Adam Terry, and Matthew Bunkley Trevathan. Advanced CAPTCHA Using Images in Sequence. USPTO Applicaton #20130031640, 2013.

[19] Robert M. French. Moving Beyond the turing Test. Communications of the ACM, 55(12):74–77, ACM, 2012.

[20] B. Fuglede and F. Topsoe. Jensen-Shannon Divergence and Hilbert Space Embedding. IEEE International Symposium on Information Theory (ISIT 2004), pages 31–31, IEEE, 2004.

[21] Philippe Golle. Machine Learning Attacks against the Asirra CAPTCHA. Proceedings of the 15th ACM conference on Computer and communications security (CCS 2008), pages 535–542. ACM, 2008.

[22] Jonathan Holman, Jonathan Lazar, Jinjuan Heidi Feng, and John D'Arcy. Developing Usable CAPTCHAs for Blind Users. Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility (ASSETS 2007), pages 245–246. ACM, 2007.

[23] Yoshifumi Kamoshida and Hiroaki Kikuchi. Word Salad CAPTCHA - Application and Evaluation of Synthesized Sentences. The 15th International Conference on Network-Based Information Systems (NBIS 2012), 0:799–804, 2012.

[24] Taku Kudo. Mecab : Yet Another Part-of-speech and Morphological Analyzer. http://mecab.sourceforge.net/.

[25] Taku Kudo and Yuji Matsumoto. Japanese Dependency Analysis Using Cascaded Chunking. Proceedings of the 6th Conference on Natural Language Learning (NLL 2002), pages 63–69, 2002.

[26] Jonathan Lazar, Jinjuan Feng, Tim Brooks, Genna Melamed, Brian Wentz, Jon Holman, Abiodun Olalere, and Nnanna Ekedebe. The Soundsright CAPTCHA: an Improved Approach to Audio Human Interaction Proofs for Blind Users. Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems (CHI 2012), pages 2267–2276. ACM, 2012.

[27] Christopher Liam. System and Method for Delivering a Human Interactive Proof to the Visually Impaired by Means of Semantic Association of Objects. USPTO Applicaton #20120232907, 2012.

[28] Chin-Yew Lin and Franz Josef Och. Orange: A Method for Evaluating Automatic Evaluation Metrics for Machine Translation. Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004), No. 501, Association for Computational Linguistics, 2004.

[29] Preslav Nakov, Francisco Guzmán, and Stephan Vogel. Optimizing for Sentence-level BLEU+1 Yields Short Translations. Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012), pages 1979–1994, Association for Computational Linguistics, 2012.

[30] Mir Tafseer Nayeem, Md. Saddam Hossain Mukta, Samsuddin Ahmed, and Md. Mahbubur Rahman. Use of Human Cognition in HIP Design Via EmotIcons to Defend BOT Attacks. IEEE 15th International Conference on Computational Science and Engineering, 0:178–185, IEEE, 2012.

[31] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL 2002), pages 311–318. Association for Computational Linguistics, 2002.

[32] G.E. Rawlinson. The Significance of Letter Position in Word Recognition. University of Nottingham, 1976.

[33] K. Saberi and DR Perrott. Cognitive Restoration of Reversed Speech. Nature, 398(6730):760, NPG, 1999.

[34] Sajad Shirali-Shahreza and M. Hassan Shirali-Shahreza. Accessibility of CAPTCHA Methods. Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence (AISec 2011), pages 109–110. ACM, 2011.

[35] Jennifer Tam, Jiri Simsa, Sean Hyde, and Luis von Ahn. Breaking Audio CAPTCHAs. Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Advances in Neural Information Processing Systems 21, pages 1625–1632. MIT Press, 2008.

[36] Luis von Ahn, Manuel Blum, Nicholas J. Hopper, and John Langford. CAPTCHA: Using Hard AI Problems for Security. Proceedings of EUROCRYPT, volume 2656 of LNCS, pages 294–311. Springer-Verlag, 2003.

[37] H. Wimmer and J. Perner. Beliefs about Beliefs: Representation and Constraining Function of Wrong Beliefs in Young Children's Understanding of Deception. Cognition, 13(1):103, ELSEVIEWER, 1983.

[38] Pablo Ximenes, Andre Santos, Marcial Fernandez, and Jr. Celestino, Joaquim. A CAPTCHA in the Text Domain. On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops, volume 4277 of LNCS, pages 605–615. Springer-Verlag, 2006.

[39] Jeff Yan, Ahmad Salah, and El Ahmad. Breaking Visual CAPTCHAs with Naive Pattern Recognition Algorithms. Proceedings of the 23th Annual Computer Security Applications Conference (ACSAC 2007), pages 279–291, ACM, 2007.

[40] 森本 浩介, 片瀬 弘晶, 山名 早人. N-gram と離散型共起表現を用いたワードサラダ型スパム検出手法の提案. 情報処理学会研究報告. データベース・システム研究会報告, 148:1–8, 情報処理学会, 2009.

[41] 山本 匠, J.D. Tygar, 西垣 正勝. 機械翻訳の違和感を用いた CAPTCHA の提案. 情報処理学会研究報告. CSEC (コンピュータセキュリティ). 2009(37):1–8, 情報処理学会, 2009.

[42] 上原 章敬, 鈴木 徳一郎, 山本 匠, 西垣 正勝. 4 コマ漫画 CAPTCHA の検討. 情報処理学会研究報告. CSEC (コンピュータセキュリティ), 2011(13):1–8, 情報処理学会, 2011.

[43] 福岡 千尋, 西本 卓也, 渡辺 隆行. 音韻修復効果を用いた音声 CAPTCHA の検討. 電子情報通信学会技術研究報告. WIT (福祉情報工学), 108(332):83–88, 電子情報通信学会, 2008.

[44] 西本 卓也, 松村 瞳, 渡辺 隆行. 音声 CAPTCHA システムにおける削除法と混合法の比較. 電子情報通信学会技術研究報告. WIT (福祉情報工学), 109(260):55–60, 電子情報通信学会, 2009.

[45] 鴨志田 芳典 菊池 浩明. 文章合成の不自然さの評価と応用. ファジィシステムシンポジウム講演論文集. 26:1069–1074, 日本ファジイ学会, 2010.

A 研究業績

学術論文誌

[1] 山口 通智. 人間ロボット判別テストのバリアフリー化のためのネット上文章の採取加工技法. ヒューマンインタフェース学会論文誌バーバルインタフェース・インタラクション特集号, Vol.15, No.4, pages 337–352, 日本ヒューマンインタフェース学会, 2013.

国際会議投稿論文

[2] Michitomo Yamaguchi, Toru Nakata, Takeshi Okamoto. An Accessible CAPTCHA system for People with Visual Disability — Generation of Human/Computer Distinguish Test with Documents on the Net. Proceedings of the 16th International Conference on Human-Computer Interaction (HCI 2014), Springer-Verlag, 2014.

国内研究会投稿論文など

[3] 山口 通智, 左瑞 麟, 岡本 健, 岡本 栄司. 署名者の追加が容易な *k*-out-of-*n* リング署名. 2013 年暗号と情報セキュリティシンポジウム予稿集 (SCIS 2013), 2A2-4, 情報処理学会, 2013.

[4] 山口 通智, 中田 亨. 人間ロボット判別テストのバリアフリー化のための言語的作問技法. 情報処理学会研究報告, CSEC (コンピュータセキュリティ), 2013(30):1–8, 情報処理学会, 2013.

[5] 山口 通智, 中田 亨, 岡本 健. インターネット上に湧出する文章の特徴とそのチューリングテストのバリアフリー化への利用. 第 12 回情報科学技術フォーラム (FIT 2013), K-049, 情報処理学会, 2013.

[6] 山口 通智, 岡本 健. 人間ロボット判別テストのバリアフリー化のための言語的作問とその自然文生成技法. コンピュータセキュリティシンポジウム 2013 (CSS 2013), 3D3–3, 情報処理学会, 2013.

[7] 山口 通智, 中田 亨, 岡本 健. ユーザ認証での画像視認テストを代替する言語的テスト. 第 39 回感覚代行シンポジウム (SSS 2013), No. 4, 感覚代行研究会, 2013.

[8] 岡本 健, 山口 通智, 三宅 輝久, 石塚 和重, 野口 栄太郎, 大越 教夫. バリアフリーな CAPTCHA の基盤構築： 視覚に障害をもつ医療系学生を事例として. 2014 年暗号と情報セキュリティシンポジウム予稿集 (SCIS 2014), 4B2-1, 情報処理学会, 2014.

[9] 岡本 健, 山口 通智, 三宅 輝久, 石塚 和重, 野口 栄太郎, 大越 教夫. 視覚に障害をもつ医療系学生に適する情報セキュリティ技術. 筑波技術大学テクノレポート, Vol.20(2), 筑波技術大学, 2014.

[10] 山口 通智, 岡本 健. 文意や文脈の解釈問題を用いた視覚障害者向け CAPTCHA とその評価. 筑波技術大学テクノレポート, Vol.20(2), 筑波技術大学, 2014.

B　CAPTCHA の関連研究

高度な認知能力を利用する方式

現在最も広く普及している CAPTCHA は、図 1.2 のように、歪んだ画像に挿入された文字列を解釈する方式である。しかし、この方式に対しては、Bursztein ら [14] による攻撃をはじめとする様々な攻撃方法が研究されており、その危殆化が問題になっている。そこで、人間のより高度な認知能力を利用する CAPTCHA が提案されている。

提示された画像の意味を解釈する CAPTCHA の代表例には、Asirra[17] がある。Asirra は、様々な動物の画像から、犬や猫などの特定の動物を選択させる方式である。この種の問題はロボットにとって非常に難しいと考えらており、多くの類似方式 [18,30,42] が提案されている。しかし Golle[21] は、犬とボールのように、特定の動物に関係の深い特徴的なオブジェクトを抽出することで Asirra に対する攻撃を成功させた。

音声の意味を解釈する CAPTCHA については、動物の鳴き声やブザー音などの特定音声からその発生源を問うもの [26] や、音声の断片から全体を推測し聞き取りをさせるもの [43,44] が提案されている。

これら画像や音声の意味を解釈させる方式は、問題として提示する関連画像や音声の準備という難点がり、新規未使用の問題を大量に作り出すことには限界がある。特に音声は、画像に比べ種類が少ないため、問題新規性要件を満たすのは難しい。

バリアフリーな方式

Holman ら [22] は、サイレンや鳥などの身近な事物を画像と音声の両方で提示し、そのいずれによっても解答可能とする方式を提案した。ただし、事物の用意の手間に鑑みると、問題新規性要件を十分に満たすことは難しい。

人間のもつ知識に依存したクイズを用いる方式 [38] も提案されている。しかし、IBM の Watson や Apple の Siri といった自然言語で質問を受け付け、正しい解答をする人工知能の登場により、この方式は無力化しつつあるとの意見 [19] がある。

文意文脈解釈問題の研究は、人工知能や認知科学の分野で古くから研究課題になっていた。

認知科学における「サリーとアン課題」[37] は、自分の知識の範囲と他者のそれとを区別できるかを問うテストが有名である。これも計算困難な問題であり、識別性要件を満たすだろう。

ディックの SF 小説 [15] においては、「このカバンは官給品なんだ。赤ん坊の皮でできている。」などと、異様な内容の文章を聞かせ、異様部分に対する身体的・感情的反応の時間遅れを計測する “Voigt-Kampff test” のアイデアが示されている。文意や文脈の理解と常識発揮の能力差を利用する例としては、先駆的なものと言えよう。

文意文脈理解能力に関する研究も盛んにおこなわれている。山本ら [41] は、人間が作った文章と機械翻訳により生成される文章との間で、人間が感じる違和感を CAPTCHA に利用した。同様に、鴨志田ら [23] の、人間が作った文章とマルコフ連鎖により自動合成された文章の比較の研究がある。Christopher[27] は、複数の文の中から内容が関連するものとしないものを選択させる方式を提案した。

文意文脈理解能力に関する既存研究の問題は、問題新規性要件に関する検討が十分でない点であ

る。これらの方式では、 検索による攻撃を避けるため公開文章を使用しないので、作問に利用可能な文章量に限界がある。

C　開発環境

図 D.1 に、4.1 節で使用した評価プログラムの外観を示す。また、評価用プログラムの開発環境を次に示す。

- Windows7 64bit 版
- VisualStudio 2012 for Desktop （Express） # 開発言語は C# を使用した。
- 外部ライブラリ
  - .NET Framework 4.5（CLS）
  - MeCab（形態素解析）
  - Cabocha（係り受け解析）
  - Saezuri（MeCab, Cabocha 等のラッパ）
  - SGMLReader（HTML → XML 処理関連のライブラリ）
  - Microsoft Speech Platform（音声合成・解析）
  - GAPI（Google API）
- 外部サービス
  - Excite 翻訳（翻訳）
  - Weblio 辞書（翻訳, シソーラス）
  - Bing（検索エンジン）
  - Google（検索エンジン）
  - Yahoo! Japan（検索エンジン）

なお、評価プログラムには、Android OS 上で動作するデモ用の簡易版も存在する。こちらは、Nexus 7 (Android OS Ver. 4.3) と Android Developer Toolkit Build: v22.0.5-757759 を利用して実装した。

D　作問例

4.1 節で実際に使用した問題の一部を記載する。ワードサラダ文識別テスト、機械翻訳文識別テスト、共通話題識別テストの作問例を、それぞれ図 D.2、図 D.3、図 D.4 に示す。

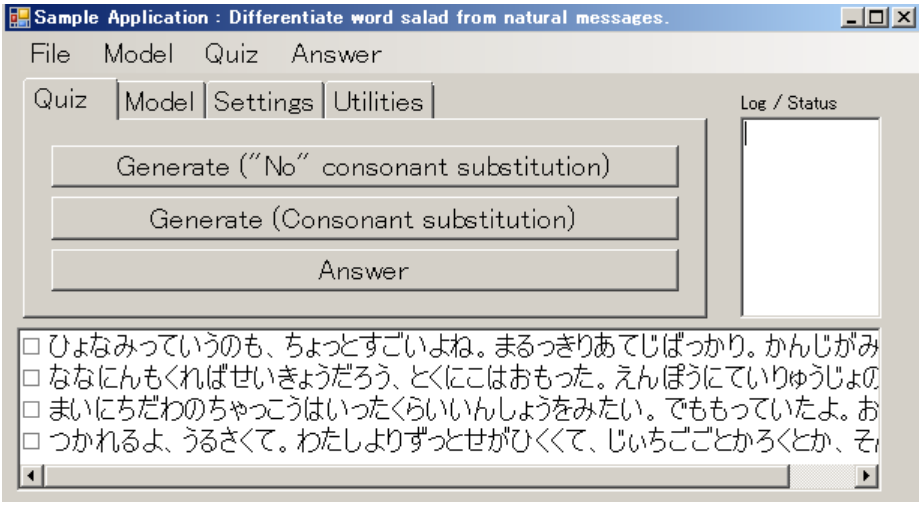


Fig. D.1　Sample Application.

ワードサラダ文識別テスト（子音交替なし）# 高い正答率の例

(1) 雨だった。その雨を部屋の窓から見ながら、邦子は、すべてをほぼあきらめた。そして、

(2) 西野だということに、やがて気づいた。美代子はすでに西野から離れて久しく、三枝子も

(3) 言うよりも、なんだろう、いちばん若い仲間という感じ。私がいっしょにいて当然なのに

(4) 直視したのをうちに砂の持ちなさいとボールには化粧してとしいて着ているのかの脚見えた名前なのか

正解: (4)

ワードサラダ文識別テスト（子音交替なし）# 低い正答率の例

(1) 出来かた持ってた新品ちがうから数分間のところから水にいまおなじ食事しˆ「から持っ横たわった

(2) していて、なにをしていても常にどことなく品があった。そしてふと視線を伏せるときの

(3) 人は話をした。恵理子がきれいに初対面の挨拶をし、小夜子は心から感心している表情で

(4) 恋人どうしになるのだ。あの少年は、理想の人なのだ。ついにあらわれた。出現した、出会っ

正解: (1)

ワードサラダ文識別テスト（子音交替あり）# 高い正答率の例

(1) あゆいていくばめんできゃ、はいごにいちだいのぴあのによるむちょうのおんがくがものしづか

(2) ちょくしぼを、うだれたにみれたまつからあいこだった。どてていっ。ぺんろみえた。そして

(3) ぎゃべりかただけでぶけど、なんちぇすてきなひとだろうと、あのときわたしはおもったの

(4) あいだで、かれじゃすそしだけまよった。それとれをてみとったかれは、どちらをもとおやま

正解: (2)

ワードサラダ文識別テスト（子音交替あり）# 低い正答率の例

(1) 41 ごうびぇんひゅだいどころ。さよこ、ははおやげてきたがたこ、いいをぼくあがっていた

(2) もとにもどり、げんひゃぎ　はかんざきけいこだった。かのこはいつものみずぎをきていた

(3) それぷつづけてかた、ようすけはえりこにあいずした。たかいふらいをあよほうこうへ、

(4) いちぶぶんとして、ちょうひょりをおよぐちとたちやすいちゅうえあろびぐすのためのぷーる

正解: (1)

Fig. D.2　Samples of Markov-chain Phrase Test.

機械翻訳文識別テスト（子音交替なし）# 高い正答率の例

(1) L を求めるには次の式を用いる。

(2) 病院の責任問題を危惧してうさこの帝王切開手術を中止させようとした。

(3) ウィリアム・ジョーンズ・ブーンという名の息子は、さらに監督教会の中の上海の監督として役立ちました。

(4) 原子英之（2009 年〜、サザンオールスターズ応援団・青森支部長）

正解: (3)

機械翻訳文識別テスト（子音交替なし）# 低い正答率の例

(1) 倉橋啓太郎 - 寺島進 （第 1 話、第 9 話・第 11 話回想）

(2) 小都羽総合病院皮膚科医師。モジャモジャ頭のアフロヘアーと体型から、

(3) 病院初診受付（クラーク）。MR の山崎に買収されやすい。人事異動で、売店配属に。

(4) 今日、神聖な名前僧院の 16 人の姉妹が聖ベネディクトのルールによって生きています。

正解: (4)

機械翻訳文識別テスト（子音交替あり）# 高い正答率の例

(1) このおーとぢょーぬのちりてきなききぃちゃつかいのこりぎぇす。それのかくちょうにより

(2) 4/28 はあさやひゃふぁんのゆうしがあつびゃり、じゅえんのおくのあきらふとしとともに

(3) ばんぐみではさいばーえーじぇんとがきょうりょくしており、げきゆうのぶもぐ

(4) しかし、じぶんじしんからいっしょへんめいざなぼうとすぎゅしせいや、まちちゅうで

正解: (1)

機械翻訳文識別テスト（子音交替あり）# 低い正答率の例

(1) この「そうぜい」は、ぶんぢぃん（しょうせつか・じじん・かじん・はいじん・ちょさくか

(2) ぢょうせつ『さんごくしえんぎ』でもとうじょうするが、ここではむちょうなしょうぐん

(3) とやまみさお・もりまつとしおへんちょ『てじぃこぶりくぐんへんせいそうらん』

(4) ぎりしゃ・ろーじゃのべんきのじしょおよびしんわ、1075

正解: (1)

Fig. D.3　Samples of Machine-translated Phrase Test.

共通話題識別テスト（子音交替なし）# 高い正答率の例
選択肢: (1) スポーツ、(2) 天気、(3) 経済、(4) 食事

- 税制課へのお問い合わせ。メールは専用フォームからの送信となり
- 30 日今日は雪、雨、強風で天候が悪く、山荘で天候回復を待つお客様
- 団体等監査（平成 22 年度事業対象）の結果は次のとおりです。なお
- リフティングチャージは安上がり？について。海外に住んでいる家族
- 2 月 15 日もうかりまっかあきまへんな”なんていう会話がいつ頃

正解: (3)

共通話題識別テスト（子音交替なし）# 低い正答率の例
選択肢: (1) スポーツ、(2) 天気、(3) 経済、(4) 食事

- 男女同権論を唱えて 150 年、なかなか日本社会の男性優位は崩れ
- 皆様にお届け致します。第２５回例会兼新年会開催日。平成２５年
- 【一等米】日本晴 10Kg 玄米 3、500 円全国の食味基準は滋賀
- 政策として民間金融機関では対応が困難な分野に対して財政融資、
- しております。udck_002071_1。jpg。おはようござい

正解: (2)

共通話題識別テスト（子音交替あり）# 高い正答率の例
選択肢: (1) スポーツ、(2) 天気、(3) 経済、(4) 食事

-)」や「au けーたいちゃくしんわりびき (ちゅう 3)」のごりようでしゃいんかんのつうしんちようの
- いすが 100 こもしゅっぴんされてはいないとおもうのせ、あくまでもすうじじょうのはなしですが
- _05_30_01。JPG。りえきをあげるにはどうしたらよいでしょうか？じちゅはりえきをあげることは、それほど
- いっちするぺーじ：7 けんのあいてむがひっとしました。1 ぺーじめをひょうじしています。Thereisenjoi！【みかぐぎゃ
- こうむいんをくしゃのかんり、くにのしゅっしやせいふほゆうかぶしきのばいきゃくとう

正解: (3)

共通話題識別テスト（子音交替あり）# 低い正答率の例
選択肢: (1) スポーツ、(2) 天気、(3) 経済、(4) 食事

- ねん 7 ぢゅき 24 にちたにやまぴろこさんの「こくもつのあめがふる」れす。(´д`) かんそうのえどうしよっかなー…。(´д`)
- 2013/7/15(つしぃ) ごぜん 11：45・きんきょうぼんとはのっとこい、いやいや、こいぴんくいろえとってもきれいです^
- しょうかいしておりまく。ひ、き、どようひぃにしかたべられないようびげんていのげきうまていしょくです。はんつくのめだまやき
- 5 つき 12 にちきのうはひさしきゅみにまとまっやあめでしたね。あめののちはくうきがすんで、けしきがどりくっき
- だんけつしてたたかっていまぬ。ぎぃなさんそときのたいいくたいかいうんどうかいはどんなものだった

正解: (2)

Fig. D.4　Samples of Topic Detection Test.