DOCTORAL THESIS

Development of a Digital Corpus and Core Language Technologies for the Ainu Language

by

Karol Nowakowski

KITAMI INSTITUTE OF TECHNOLOGY
GRADUATE SCHOOL OF ENGINEERING

September 2020

Acknowledgements

This work would be not be possible without the support that I received from many people. I would like to reserve this page to thank them.

I want to thank my supervisor, Professor Fumito Masui, for believing in me and for providing guidance and feedback throughout this project. His insight and knowledge into the subject matter helped me a lot in all aspects of my studies. I want to thank Professor Michal Ptaszynski, who invited me to take part in the research project devoted to the Ainu language and introduced me to the fascinating world of Natural Language Processing. He provided extensive support and mentorship in my academic development and served as a constant source of expertise and encouragement. I would like to thank Professor Yoshio Momouchi for contributing annotated Ainu language materials used in this study and useful advice. I am also thankful to Professor Kyōko Murasaki and Ms. Rueko Ōsuga, for their willingness to share their time, materials and extensive knowledge about the Ainu language. I would like to express my sincere gratitude to all participants of the surveys conducted in the course of this research, who took the time to share their valuable feedback. I gratefully acknowledge the funding received from the Japanese Ministry of Education, Culture, Sports, Science and Technology to undertake my PhD studies at the Kitami Institute of Technology. Finally, I would like to say a heartfelt thank you to my family and friends, for all the help and support they have provided. Special thanks to my wife Jagna, for her patience and encouragement.

Karol Nowakowski
September 2020

ABSTRACT

Technologies being developed within the field of Natural Language Processing (NLP) have an important role to play in the urgent tasks of documenting, analyzing and revitalizing endangered languages. At the same time, the rapid development and the spread of language-related technologies observed in recent decades may result in creating a technological gap between smaller languages and majority languages – which in turn will threaten the survival of the former group – if linguistic minorities are not provided with equal access to said technologies. Among such languages is Ainu, a critically endangered language isolate native to northern parts of the Japanese archipelago. In this dissertation, I present the results of my research devoted to the development of Natural Language Processing technologies for the Ainu language.

I begin with presenting the background of this research. In particular, I provide an overview of previous research in the field of Natural Language Processing for under-resourced and endangered languages. I then describe the characteristics and current situation of the Ainu language, and review some of the related research on the Ainu language, including the few existing studies in the field of Natural Language Processing. I also describe the major challenges facing Ainu language research and revitalization efforts.

After that, I report the results of a survey conducted on a group of Ainu language learners, scholars, and other people interested in the Ainu language, with the aim of evaluating their experiences and needs concerning language-related technologies. Based on the findings from the questionnaire, I outline future research goals.

A major obstacle for the application of advanced language processing technologies to minority languages, including Ainu, is the lack of high-volume digital linguistic resources, such as text and speech corpora and in particular, annotated corpora. One of the main contributions of the presented research is the compilation of a large-scale corpus of Ainu. It covers a wide range of documents in a consistent structure, allowing for the application of corpus-driven approaches to NLP, such as statistical language models and representation learning techniques.

The above-mentioned corpus is then utilized in the development of Natural Language Processing tools for Ainu, namely word segmentation models and part-of-speech taggers. For the first task, I propose a novel algorithm: MiNgMatch Segmenter. I argue that the problem of identifying word boundaries can be reduced to finding the shortest sequence of lexical n-grams matching the input text, thus reducing its computational cost and increasing efficiency of the process. I perform a series of experiments comparing the algorithm with systems utilizing state-of-the-art statistical language modelling techniques, as well as a neural model performing word segmentation as character sequence labelling. The experimental results demonstrate strong performance of the proposed approach, comparable with the other best-performing models. Next, I apply a

state-of-the-art generator of sequential taggers – SVMTool – and a tagger based on Artificial Neural Networks, equipped with word representations inferred from the corpus, in the task of automatic part-of-speech annotation. Evaluation results reveal that they perform better than the dictionary-based system proposed in previous research (POST-AL), especially when applied to out-of-domain data.

Furthermore, I describe a preliminary Ainu language conversational program for the Pepper robot, which serves as a proof of concept of how robots could support Ainu language education. The proposed robot can hold simple conversations, teach new words and play interactive games using the Ainu language. In a group of Ainu language experts and experienced learners whom I asked for feedback (in the form of a survey study), the majority supported the idea of developing an Ainu-speaking robot and using it in language teaching.

Advances in cross-lingual learning indicate that the problems facing Natural Language Processing for low-resource languages can be, to a certain extent, alleviated by transferring knowledge from resource-rich languages. Unsurprisingly, however, such techniques tend to yield the best results for closely related languages, whereas the Ainu language is a language isolate, with no known cognates. Nevertheless, given the similarity of phonological systems and some grammatical constructions between Ainu and Japanese, it may still be beneficial to use the existing Japanese resources as a starting point in the development of language processing technologies for Ainu. In this thesis, I describe two preliminary experiments in cross-lingual knowledge transfer from Japanese to Ainu: firstly, I propose a method for generating useful word representations by using an Ainu-Japanese parallel corpus with morpho-syntactic annotations on the Japanese side. When applied to the problem of part-of-speech tagging of Ainu text, the proposed method contributes positively to the performance of a neural tagger. Secondly, I investigate the performance of Japanese models for Speech Synthesis and Speech Recognition in generating and detecting speech in the Ainu language. I perform human evaluation of the synthesized speech in terms of intelligibility and pronunciation, as well as automatic evaluation of Speech Recognition. Experiment results suggest that cross-lingual transfer from Japanese has a potential to facilitate the development of speech technologies for Ainu, especially in the case of Speech Recognition.

ABSTRACT IN JAPANESE (論文内容の要旨)

「自然言語処理」（NLP）研究分野は，消滅の危機に瀕している少数言語の記録保存や分析，再活性化といった喫緊の課題解決において重要な役割を担うと期待されている． ここ数十年において言語関連技術は急速に発展・普及しているが，少数言語話者に対してもそれらの技術を活用する機会が与えられなければ，多数言語と少数言語との間に技術的ギャップが生じ，後者の危機がさらに深刻化する恐れがある． これらの言語には，日本の現地語の一つであり危機言語となっているアイヌ語がある． 本論文では，アイヌ語のための自然言語処理技術開発に関する研究結果について報告する．

まず，本研究の背景を紹介する． 具体的には，自然言語処理の分野における言語資源の少ない言語および消滅危機にある言語を対象とした先行研究を簡潔に紹介する． 次に，アイヌ語の特徴と現状を説明し，自然言語処理分野における数少ないアイヌ語に関する研究を紹介するとともに，アイヌ語の研究および復興の取り組みが直面している主たる課題について説明する．

次に，アイヌ語のための言語処理技術の潜在的なユーザーの特徴や具体的なニーズを知るため，アイヌ語に関わる諸活動に携わっている人々（研究者，学習者，講師など）を対象に実施されたアンケート調査の結果を報告する． そして，アンケートの結果をもとに，今後の研究課題について論述する．

最先端の自然言語処理技術を 有効活用するためには，統一された形式を持つ大量のテキストからなるデータベース，すなわち「電子化コーパス」が不可欠である． アイヌ語についてはオンライン辞書や資料集が多少存在するものの，規模の大きな電子化コーパスは今まで存在しなかった． 本研究の主要な貢献の一つはこの問題に対処することである．筆者は，統計的言語モデルや分散表現学習など，大規模なコーパスを必要とする自然言語処理手法の応用を可能にする，186万文字からなるアイヌ語の電子化コーパスを設計し構築したことである．

続いて，構築したコーパスを単語分割モデルおよび品詞タガーといったアイヌ語のための自然言語処理技術の開発に応用した． 前者においては，MiNgMatch Segmenterという，入力文と一致する最短の単語n-gram列を探索する高速の単語分割アルゴリズムを提案，実装した． そして，提案手法の有効性を検証するために，最先端の統計的言語モデルを利用した単語分割器および文字単位で処理を行う「文字タグ付け法」に基づくニューラルネットワークベースの単語分割器を用いて比較実験を実施した結果，提案手法が十分に有効であることが確認できた． 次に，Support Vector Machines手法を応用した最先端の品詞解析ツール（SVMTool）と，コーパスから学習した分散表現を用いたニューラルネットワークベースの品詞タガーをアイヌ語の品詞解析に応用した． 上記タガーの有効性を検証するために，先行研究で提案された辞書ベースの品詞タガーであるPOST-ALと比較する検証実験を行った結果，提案手法の性能が大幅に上回っていることが確認された．

さらに，アイヌ語教育の場面においてAIやロボットの活用可能性を論証するすることを目的として，アイヌ語の自動会話プログラムをロボットに搭載して活用するための予備実験を実施した． ロボットは，アイヌ語で簡単な会話をすることの他に，新しい単語を教えたり，アイヌ語を用いたインタラクティブゲームを実行することもできる． アイヌ語を話すロボットを開発し言語教育に応用するアイデアに関して，アイヌ語の専門家や学習者を対象にアンケート調査を実施した結果，回答者のほとんどが賛成であることが分かった．

最後に，日本語からアイヌ語への知識転移の効果について報告する． 近年における「マルチリンガル学習」の進化が示しているように，言語資源の少ない言語の自然言語処理が直面する問題は，資源が豊富な言語から知識を転移することによってある程度軽減できる． ただし，当然のことながら，そのような手法は近縁言語に対して最良の結果をもたらす傾向がある一方，アイヌ語のようないわゆる「孤立言語」には効果が小さいと考えられる． しかし，アイヌ語と日本語との音韻体系やいくつかの文法構造の類似性を考慮すると，日本語データを用いたマルチリンガル学習がアイヌ語の言語処理技術を開発する上で有効であると期待できる． 筆者は，日本語からアイヌ語への知識転移の効果を検証するために2つの実験を実施した． まず，アイヌ語-日本語対訳コーパスと日本語形態素解析器を使用し，有効な単語分散表現（単語ベクトル）を計算する方法を構築した． 構築した手法は，アイヌ語テキストの品詞解析の問題においてニューラルの品詞タガーの精度の向上に貢献することが確認された． 次に，日本語音声合成および音声認識モデルのアイヌ語に応用した際のパフォーマンスを検証した． 合成されたアイヌ語音声の理解度および発音について，人間による評価と音声認識の自動評価を行った． その結果，日本語からの言語知識の転移がアイヌ語音声技術の開発を促進する可能性があることが示唆された．

以上より，本論文で報告した，絶滅危機言語であるアイヌ語を対象とした一連の自然言語処理技術の開発研究の成果が，アイヌ語の保存復興に十分貢献するものと結論できる．

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Ainu is a language isolate native to northern parts of the Japanese archipelago, which – as a result of the Japanese and Russian colonization of the area and a subsequent policy of assimilation – has been rapidly loosing speakers in the last two centuries, to the point where it is no longer being used as a means of daily communication, and is currently recognized as nearly extinct (e.g., by Lewis et al. [71]).

At the same time, researchers have been collaborating with native speakers to document the language, and, especially in the last few decades, multiple initiatives have been undertaken by the members of the Ainu community to preserve their mother tongue and promote it among the young generations. Examples include Ainu language courses offered throughout Hokkaidō, a radio course ("Ainugo Rajio Kōza"[1]) broadcast by the STV Radio in Sapporo, and an annual Ainu language speech contest, "Itak an ro" ("Let's speak Ainu!").

I strongly believe that the field of Natural Language Processing (NLP) has an important role to play in the urgent tasks of documenting, analyzing and revitalizing endangered languages, such as Ainu. This dissertation presents my research aimed at the development of Natural Language Processing tools for the Ainu language.

[1] https://www.stv.jp/radio/ainugo/index.html

1.1 Structure of the Thesis

The remainder of this thesis is organized as follows. Chapter 2 describes the background of my research. In particular, I review some of the related research in the area of Natural Language Processing for under-resourced languages. After that, I briefly describe the characteristics and the current status of the Ainu language. I also provide an overview of some of the previous studies on the Ainu language, including the few existing research projects in the field of Natural Language Processing.

In Chapter 3, I report the results of a survey conducted on a group of Ainu language learners, scholars, and other people interested in the Ainu language, with the aim of evaluating their experiences and needs concerning language-related technologies. Based on the findings from the questionnaire, I outline a general framework for Ainu language research involving Natural Language Processing techniques, including future research goals.

Chapter 4 presents the first large-scale digital corpus of the Ainu language, compiled in the course of this research.

In Chapter 5, I discuss the results of my experiments in developing Natural Language Processing tools for Ainu, including tools for three text processing tasks (transcription normalization, word segmentation and part-of-speech tagging) and a rule-based dialogue agent in the form of a robot.

Chapter 6 concludes the thesis, with a review of the principal findings of this research and ideas for future work.

Chapter 2

Background

2.1 Natural Language Processing for Endangered Languages

UNESCO estimates that at least half of the languages currently used around the world are losing speakers and about 90% of them may be replaced by dominant languages by the end of the 21st century [149].

Technologies being developed within the field of Natural Language Processing (NLP) have a great potential to support the urgent tasks of documenting, analyzing and revitalizing endangered languages. At the same time, the rapid development and the spread of language technologies observed in recent decades may result in creating a technological gap between smaller languages and majority languages – which in turn would threaten the survival of the former group – if linguistic minorities are not provided with equal access to said technologies.

For these reasons, multiple research initiatives have been undertaken in recent years with the aim of developing linguistic resources (such as lexicons [61] and annotated corpora [4, 47, 43, 3]) and speech or text processing technologies [10, 127] for under-resourced and endangered languages. Abney and Bird [2] advocated for the construction of a multi-lingual corpus in a consistent format allowing for cross-linguistic automatic processing and the study of universal linguistics. Bird and Chiang [12] discussed the potential role of machine translation in language documentation. Blokland et al. [14, 15] and Gerstenberger et al. [42, 41] proposed

the application of proven Natural Language Processing approaches as a method to facilitate language documentation efforts, in particular to automate the process of corpus annotation and to support the integration of legacy linguistic materials in contemporary documentation projects. The "Digital Language Survival Kit" [9], published as a part of the Digital Language Diversity Project, lists some of the basic resources and technologies (such as spell checkers, part-of-speech taggers, and speech synthesis and recognition tools) necessary to improve the digital vitality of minority languages.

2.2 The Ainu Language

Ainu is an agglutinating, polysynthetic language with SOV (subject-object-verb) word order. Ainu verbs are obligatorily marked with pronominal affixes (different for intransitive and transitive verbs) indicating person and number of the subject and the object [20]. Polysynthetic characteristics (such as incorporation and concentration of various morphemes in the verbal complex) are stronger in classical Ainu (the language of the traditional Ainu epics) than in colloquial language [139]. The first of the following examples demonstrates noun incorporation in Ainu. In the second one, a similar meaning is expressed without incorporation.

(A) ku-kamuy-panakte [132]

1st.person.singular.subject-god(s)-punish

"I was punished by the gods"

(B) kamuy en-panakte [132]

god(s) 1st.person.singular.object-punish

"The gods punished me"

Although numerous attempts have been made to relate Ainu to Paleo–Asiatic, Ural–Altaic, or Malayo–Polynesian languages, to individual languages spoken in the same region, such as Japanese and Gilyak, or even to such remote groups of languages as Semitic and Indo–European (see [139]), until the present day none of these hypotheses have gained wider acceptance. Thus, it is usually classified as a language isolate. On the other hand, a number of grammatical constructions

and phonological phenomena are believed to have developed under the influence of Japanese (see [20]).

There are multiple regional varieties of the Ainu language. In the past, the Ainu inhabited the Kurile archipelago, Sakhalin, Hokkaidō, and presumably also the Tōhoku region in northern Honshū and the southern part of Kamchatka. After the World War II, the Ainu from Sakhalin and the Kurile Islands were relocated to Hokkaidō [126]. There is very little data available on the Kurile dialects [20]. The dialects of Sakhalin exhibit properties not found in Hokkaidō Ainu (such as phonemic distinction between short and long vowels), the dissimilarities being large enough for many experts (e.g., [126, 90]) to describe the Hokkaidō and Sakhalin dialects as mutually unintelligible.

2.2.1 Phonology

Phonemic inventory of the Ainu language consists of five vowel phonemes: /i, e, a, o, u/, and twelve consonant phonemes: /p, t, k, c, s, h, r, m, n, y, w, ʔ/. For detailed analyses, please refer to Refsing [126], Shibatani [139] and Bugaeva [20]. At the level of phonemes, phonological system of Ainu exhibits significant overlap with that of the Japanese language. There are, however, substantial differences between the two languages in phonotactics. In Japanese the basic syllable structure is CV (C = consonant, V = vowel), with only two types of consonants that may close a syllable: a geminate (doubled) consonant, and syllable-final nasal /N/. On the contrary, in Hokkaidō Ainu all consonants except /c/, /h/ and /ʔ/ may occur in syllable coda position [139]. Furthermore, certain combinations of phonemes, such as /ye/ and /we/ are not permitted in modern Japanese or can only be found in foreign loan words (or have an irregular phonetic realization, as in the case of /tu/, pronounced as [tsɯ]), while the corresponding Ainu phonemes are not subject to such restrictions.

Although, in contrast to Japanese and Sakhalin Ainu, the opposition between short and long vowels is not distinctive in Hokkaidō Ainu [139], vowels are often prolonged in certain forms (e.g., interjections – see [94]) or at the end of a sentence (e.g., in imperative sentences – see [131]).

Both Japanese and the Ainu language have a pitch accent system, but with different characteristics: (Tokyo) Japanese exhibits a so-called “falling kernel” accent

(i.e. the change from high to low pitch is distinctive), whereas in Ainu the position of the rise in pitch is important [20]. The place of the accent in an accented word in Tokyo Japanese is not predictable without prior lexical knowledge [139]. In Ainu, apart from a few words with irregular accent, the accent falls on the first syllable if it is closed, and on the second syllable otherwise. The accent of a word may also be affected by the attachment of certain affixes [20].

Intonation in the Ainu language is falling in declarative sentences and rising in questions [126]. In Japanese, on the other hand, not all types of questions are pronounced with a rising intonation [39].

2.2.2 Transcription

Most written texts in Ainu are transcribed using Latin alphabet and/or the Japanese *katakana* syllabary. *Katakana*, in its official version, has no means to represent closed syllables not occurring in the Japanese language. As a result, it is not possible to produce phonemically accurate transcriptions of many Ainu words containing syllable-final consonants. In older documents, such syllables were usually expressed as a combination of two characters representing open (CV) syllables. For instance, in one of the oldest Ainu language dictionaries, *Ezo hōgen moshiogusa* [148], the word *apkas* (/apkas/, "to walk") was trascribed as アプカシ (/apukasi/). In contemporary transcription conventions, this problem is solved by using an extended version of the syllabary, with small-sized (*sutegana*) variants of *katakana* characters to denote syllable-final consonants (e.g., ⟨ㇰ⟩ /k/ derived from ⟨ク⟩ /ku/, and ⟨ㇷ⟩ /p/, from ⟨プ⟩ /pu/).

After two centuries of constant evolution – with multiple alternative notation methods being simultaneously in use – Ainu orthography has been, to a certain degree, standardized[1,2]. Concerning word segmentation[3], however, no standard

[1]One of the milestones in that process was a textbook compiled by the Hokkaidō Utari Association (now Hokkaidō Ainu Association) in cooperation with Ainu language scholars, published in 1994 under the title *Akor itak* ("our language") [51]. It was intended for the use in Ainu language classes held throughout Hokkaidō and included a set of orthographic rules for both Latin alphabet and *katakana*-based transcription. They are widely followed to this day, for example by Hiroshi Nakagawa [92], Suzuko Tamura [143] and the authors of the Topical Dictionary of Conversational Ainu [96].

[2]For detailed analyses of notation methods employed by different authors and how they changed with time, please refer to Kirikae [66], Nakagawa [93] and Endō [36].

[3]In Ainu in its written form, whitespaces are used to indicate word boundaries both in the

guidelines have been established to date [141, p. 198][131, p. 5], and polysynthetic verb morphology only adds to the confusion [92, p. 5]. Contemporary Ainu language experts – while taking different approaches to handling certain forms, such as compound nouns and lexicalized expressions – generally treat morphemes entering in syntactic relations with other words as distinct units, even if they are cliticized to a host word in the phonological realization[4]. The problem is most noticeable in older documents and texts written by native speakers without a background in linguistics, who tended to divide text into phonological words or larger prosodic units (sometimes whole verses) – see Sunasawa [141, p. 196]. As a consequence, orthographic words in their notation comprise, on average, more morphemes (an example is shown in Table 2.1). This, in turn, leads to an increase in the proportion of items not to be found in the existing dictionaries, which makes the content of such texts difficult to comprehend by less experienced learners. Furthermore, in the context of Natural Language Processing it renders the already limited data even more sparse. In order to facilitate the analysis and processing of such documents, a mechanism for word boundary detection is necessary.

Table 2.1: A fragment from the *Ainu shin-yōshū* [29]; original text by Chiri (top) and modernized transcription by Kirikae [65] (middle).

Nenkatausa wakka unkure
Nen ka ta usa wakka un kure
Meaning: “Someone, please give me water”

2.2.3 Current Situation

While the exact number of Ainu language speakers is difficult to determine, in a survey conducted in 2017 by the Hokkaidō regional government [50], only 4.1% out of 671 respondents answered that they were able to communicate using the Ainu language. This situation is a consequence of the language shift from Ainu to

case of Latin script and *katakana*-based transcription (which is one of the differences between how *katakana* is used for Ainu and for Japanese, where words in a sentence are not delimited).

[4]In dictionaries, study materials and written transcripts of Ainu oral tradition that were published in the last few decades [e.g., 143, 146, 128], it is a popular practice to use *katakana* to reflect pronunciation, while parallel text in Latin characters represents the underlying forms.

Japanese which started in the 19th century and resulted in the mother tongue of the Ainu people no longer being transmitted to next generations [75].

That being said, in the past few decades many Ainu people have found a new sense of pride in their ancestral culture, which resulted in increased interest in the Ainu language. One of the central figures in the movement to reclaim Ainu identity was Shigeru Kayano (1926–2006), who established the first Ainu language school in Nibutani in 1983 [79]. Today, a number of Ainu language courses are offered throughout Hokkaidō and in other regions of Japan. The Foundation for the Research and Promotion of Ainu Culture (FRPAC) publishes teaching materials for multiple dialects of Ainu, collaborates with the STV Radio in Sapporo in broadcasting a series of Ainu language courses ("Ainugo Rajio Kōza"), and holds an annual Ainu language speech contest, "Itak an ro" ("Let's speak Ainu!").

In recent years, new contexts of language use have emerged. A magazine in the Ainu language, *Ainu Taimuzu* ["Ainu Times"][5], has been published since 1997. It is a unique example of Ainu being used in the context of contemporary affairs. There are musicians singing in Ainu, such as Oki and "Marewrew"[6]. Satoru Noda's best-selling *manga* series, *Golden Kamuy*, set in early 20th century Hokkaidō, with one of the main characters being an Ainu girl, is consulted with a renowned Ainu language expert, Hiroshi Nakagawa, and regularly features dialogues in Ainu. Since 2018, Dōnan Basu (a local bus company in southern Hokkaidō) has been broadcasting on-board announcements in the Ainu language[7] on a number of bus lines operating in Biratori. Maya Sekine runs a YouTube channel[8] that teaches Ainu and introduces Ainu culture. Recently, an Ainu course including over 250 words and phrases – developed in cooperation with the Hokkaido University Center for Ainu and Indigenous Studies – has been launched on a mobile language learning platform, Drops[9].

[5] https://otarunay.at-ninja.jp/taimuzu.html
[6] http://www.tonkori.com/
[7] https://www.ff-ainu.or.jp/event/direct/2018aynu_itak_ani_a=i=siramkire_kusu_ne/index.html
[8] https://www.youtube.com/channel/UCsvS5QjLwvlVhWpK48L57Cg
[9] https://languagedrops.com/blog/learn-ainu-with-drops

2.2.4 History of Ainu Language Research

The earliest sources on the Ainu language date back to the 17th century, most of them being small wordlists compiled by travellers, missionaries, or official Japanese interpreters [126]. The 19th century brought the first dictionary publications by Japanese authors, such as Uehara and Abe [148] and Jinbō and Kanazawa [55] (Ainu–Japanese), as well as by Europeans: Dobrotvorskij [32] (Ainu–Russian), Batchelor [7] (Ainu–English–Japanese), Radliński [124] (Ainu–Polish–Latin) and others. Further development of Ainu lexicography in Japan occurred in the second half of the 20th century and resulted in publishing some of the most comprehensive dictionaries of the Ainu language (all of them compiled as Ainu–Japanese bilingual dictionaries), such as the ones by Mashiho Chiri [26, 28, 27], Shirō Hattori [48], Hiroshi Nakagawa [92], Suzuko Tamura [143], Shigeru Kayano [60], and Hideo Kirikae [65].

Another important branch of Ainu language studies is the documentation and study of the Ainu people's oral literature. One of the pioneers in this field was a Polish anthropologist, Bronisław Piłsudski, who spent several years in Sakhalin between 1886 and 1905, studying Ainu language and culture, and in 1912 published a collection of 27 Ainu texts with English translations and comments. He also produced the earliest known sound recordings of the Ainu language, dating from 1902–1903 [76].

A latter example, and probably one of the best known studies concerning the Ainu oral tradition are the works of Kyōsuke Kindaichi (e.g., [63, 64]), who devoted his research to translating and analysing the *yukar* epics.

Similar studies were also undertaken by Yukie Chiri (native Ainu who compiled a collection of 13 *yukar* stories: the *Ainu shin-yōshū* [29], first published in 1923) and Shigeru Kayano [59], among others.

2.3 Challenges Facing Ainu Language Research and Revitalization

As a result of the language shift from Ainu to Japanese, there are currently very few people capable of speaking the Ainu language. Even among language

instructors, some people admit that they could not hold a natural conversation in Ainu[10]. Furthermore, due to logistic and human resource constraints, Ainu language classes are only held in a limited number of larger cities, sporadically (once a week or even less frequently), and often only seasonally[11]. All this means that the access to Ainu language education is severely limited.

The problem of limited resources applies to Ainu language research, too. Even though it is no longer possible to collect new fieldwork data [18, p. 247], decades-old recordings and handwritten manuscripts still await analysis and publication[12], and linguistically annotated materials are extremely rare.

Another problem is the scarcity of language resources in digital format, allowing for the application of modern corpus linguistics and computational linguistics techniques. While being relatively well documented in comparison to other endangered languages [20, p. 464], Ainu is not currently being used by the Ainu community as the primary language of communication or as the official language of any institutions. As a consequence, it is not possible to gather large amounts of data by crawling the Internet, or to use the Web as a large-scale online corpus [see 62], as can be done for many other languages, including less-studied ones.

A possible solution is to digitize legacy materials (many of which have the additional advantage for linguistic research, that they date back to the period when there were still many fully proficient speakers of Ainu). The problem with older texts, however, is that until the last decade of the 20th century there existed no widely-accepted, standard guidelines for transcription of the Ainu language (see Section 2.2.2). This makes it more difficult to perform automated text analyses spanning multiple data sets, and in the context of Natural Language Processing it renders the already limited data even more sparse.

At present, Ainu vocabulary does not cover many modern concepts, such as technology or politics. This is a major barrier for promoting active use of the Ainu language in present-day society. For revitalization efforts to succeed, a systematic

[10] Information from personal communications with teachers of the Ainu language and members of Ainu community. See also [79, pp. 74–75].

[11] As an example, the 2019 edition of the beginner level Ainu language course run by the Foundation for Ainu Culture (https://www.ff-ainu.or.jp/) was held in three cities in Hokkaidō (Sapporo, Kushiro and Shiraoi), with up to 20 lessons over the course of 6 or 8 months.

[12] Examples include the speech data stored by the Ainu Museum in Shiraoi, amounting to ca. 700 hours (see: http://ainugo.ainu-museum.or.jp/pages/about_media.html).

approach to introducing new lexical items – and ensuring their conformity with Ainu phonology and morphology – will need to be developed [79, 54].

Another aspect that must be taken into account in Ainu language research and revitalization activities, is dialectal variation. Unlike in state-level languages, such as Japanese, there is no "standard Ainu", which means that studying the Ainu language is equivalent to studying one or more regional variants of it.

2.4 Previous Research in Natural Language Processing for Ainu

Studies devoted to the Ainu language in the field of Natural Language Processing are few in number. From 2002, Momouchi and colleagues have been working on preparing ground for an Ainu–Japanese machine translation system. In the first stage of their research they tried to develop methods for automatic extraction of word translations based on a small parallel corpus [57, 85, 34]. Later, Azumi and Momouchi [5, 6] started the development of tools for analysis and retrieval of hierarchical Ainu–Japanese translations. Momouchi, Azumi and Kadoya [86] annotated a subset of the *yukar* epics included in the *Ainu shin-yōshū* [29] with information such as parts of speech, Japanese translations and normalized transcription, with the intent of using it for the development of a machine translation system. Lastly, Momouchi and Kobayashi [87] compiled a dictionary of Ainu place names and used it to create a system for the analysis of Ainu topological names. A more recent project, by Senuma and Aizawa [134, 135], aims at creating a small dependency treebank in the scheme of Universal Dependencies. Furthermore, Matsuura et al. [80] reported the development of an end-to-end Speech Recognition model for Ainu.

In 2012, Ptaszynski and Momouchi [120, 119] developed POST-AL, the first automatic part-of-speech tagger for the Ainu language, with additional functionalities of word segmentation, morphological analysis and word-level translation. It was trained using a dictionary compiled by Kirikae [65] and performed disambiguation based on lexical n-grams obtained from sample sentences included in the dictionary. Later, Ptaszynski et al. [118] implemented the above-mentioned functionalities in one coherent toolkit. It two later studies, Ptaszynski, Nowakowski, Momouchi and

Masui [122] and Ptaszynski et al. [117] investigated the possibility of improving the system's performance in part-of-speech tagging and word segmentation, by testing it with four different dictionaries. Nowakowski, Ptaszynski and Masui [102, 107] achieved further improvements by using a combination of two dictionaries instead of one and applying a hybrid method of part-of-speech disambiguation, based on n-grams and word frequency, as well as a segmentation algorithm maximizing mean token length. Nowakowski et al. [105] applied the SVMTool – a state-of-the-art generator of sequential taggers based on Support Vector Machines – to the task of part-of-speech tagging, with good results.

Apart from the development of tools for automatic processing of the Ainu language, Nowakowski, Ptaszynski and Masui [101] launched a project to create a large-scale annotated corpus of Ainu. Utility of the corpus for NLP applications was positively verified by Nowakowski, Ptaszynski and Masui [104, 103], who used it to train an n-gram language model-based word segmentation system.

As for studies employing modern digital technologies in a wider sense, in recent years there has been a number of research projects focused on creating online dictionaries and repositories of materials in the Ainu language (texts with translations, as well as voice and video recordings), such as the ones by the National Institute for Japanese Language and Linguistics [96], Chiba University Graduate School of Humanities and Social Sciences [24], and The Ainu Museum [145].

Chapter 3

Framework for Ainu Language Processing Research

In this chapter, I present the results of a survey conducted among a group of people involved in activities associated with the Ainu language, in order to identify their needs with regard to language-related technologies. Analysis of the survey serves me as a point of departure for setting out a road map for future work.

Large portions of this chapter have appeared in [106].

3.1 Survey on Technologies Needed in Ainu Language Research and Revitalization

In this section I present the interim results of an ongoing survey study to assess the needs for language technologies for studying and revitalization of the Ainu language. The target group of the survey are all the people engaged in activities related to the Ainu language, such as learners, instructors, linguists and researchers of Ainu literature and/or culture[1].

3.1.1 Survey Design

The survey consists of the following six questions:

[1]The survey can be found at: https://forms.gle/PU92EE55qAz15Dc8A

1. In what context do you have contact with the Ainu language?

   Options: as a linguist; as a researcher of Ainu literature and/or culture; as a learner of the Ainu language; as an Ainu language instructor.

2. How long have you been interested in the Ainu language?

   Options: less than 1 year; 1-5 years; more than 5 years.

3. Choose Ainu dialects you are primarily interested in.

   Options: Chitose; Shiraoi; Horobetsu; Mukawa; Saru; Shizunai; Samani; Tokachi; Bihoro; Kushiro/Shiranuka; Ishikari-gawa; Sakhalin Ainu; Kuril Ainu; all dialects.

4. Do you use the following language-related technologies for other languages you are using or studying?

   Options: online dictionaries; mobile dictionaries; spell checkers; machine translation; speech recognition; speech synthesis; part-of-speech taggers; morphological analyzers; syntactic parsers; digital corpora.

5. Do you use the following language-related technologies for Ainu?

   Options: online dictionaries and text collections; Latin script to *kana* conversion program[2].

6. If the following technologies were available for the Ainu language, which of them would you like to use?

   Options: mobile dictionary; spell checker; machine translation; speech recognition; speech synthesis; part-of-speech tagger; morphological analyzer; syntactic parser; digital corpus.

In questions 1 and 3 through 6, respondents are allowed to add their own answers, apart from the preset answer options.

3.1.2 Results

Up to the time of writing this chapter (2020/02/20), the survey has been taken by 13 respondents. Although it is not a sufficient sample to be considered as a

[2] http://aynuitak.at-ninja.jp/WEBhenkan/chiyu.htm

reliable, statistically significant source of information about the entire community, the respondents so far comprise in majority a valuable sample of long-time Ainu language learners and researchers, revealing useful insights, which I present below.

3.1.2.1 Analysis of the Survey Respondent Population

Figure 3.1 presents the distribution of respondents by the type of Ainu language-related activities they are primarily involved in. More than half of the participants are learners of Ainu, while the second largest group are researchers from the field of linguistics. Unfortunately, at this point I have only received a single response from Ainu language instructors.

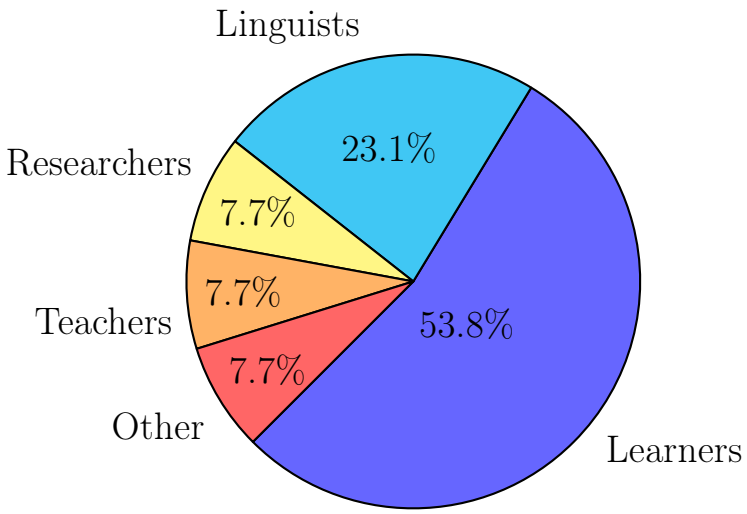

Figure 3.1: Distribution of respondents by the type of Ainu language-related activities

As shown in Table 3.1, the majority of the participants have over 5 years of experience with the Ainu language.

With regard to dialects of Ainu, 5 participants (38.5%) responded that they were interested with all the regional varieties. Individual dialects most popular among the respondents are the Kushiro/Shiranuka dialect (38.5%), the Saru dialect (30.8%) and the dialects of Sakhalin (30.8%).

Table 3.1: Distribution of respondents by experience with the Ainu language (in years)

Years:	**Respondents (%):**
> 5	9 (69.2%)
1-5	3 (23.1%)
< 1	1 (7.7%)

3.1.2.2 Experience with Language-related Technologies

In the second part of the survey, I asked the participants about their experience with language-related technologies. Figure 3.2 shows the numbers of participants using different types of technologies for languages other than Ainu. Most respondents are acquainted with online dictionaries, and almost half of them are also using mobile dictionary applications. The third option with the highest number of selections are digital corpora, which are used by all the three linguist participants, the person engaged in Ainu language teaching and one learner. None of the participants responded that they were using speech technologies or part-of-speech taggers, morphological analyzers and syntactic parsers.

As for the few technologies available for the Ainu language (see Table 3.2), the majority of the participants are users of online dictionaries and/or digital text collections. Nearly one third of them also have experience with the automatic converter of Ainu text written in Latin alphabet to *kana* syllabary.

Table 3.2: Numbers of respondents acquainted with existing language technologies for Ainu

Technology:	**Respondents (%):**
Online dictionaries and text collections	9 (69.2%)
Latin script – *kana* conversion program	4 (30.8%)

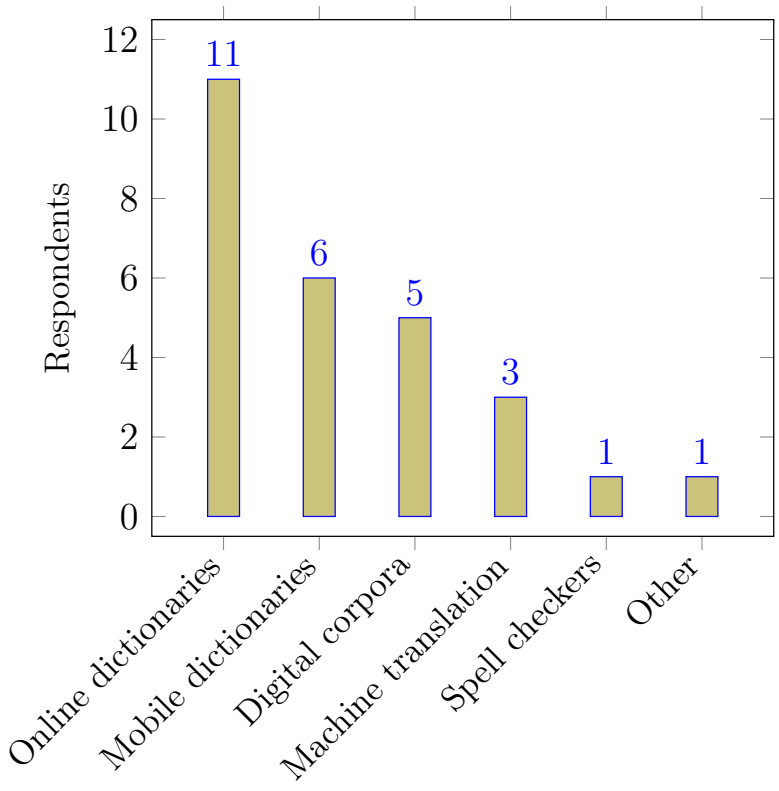

Figure 3.2: Numbers of respondents acquainted with language-related technologies

3.1.2.3 Expectations about Language-related Technologies for Ainu

Answers to the survey's final question are summarized in Figure 3.3. The leftmost (plain blue) column in each cluster corresponds to the total number of participants interested in using the particular technology, whereas the remaining columns represent the results separately for three different groups of respondents (depending on their answer to Question 1): learners, linguists and language instructors. The white portion of each bar shows the number of participant from the respective group who did not select the given answer.

3.1.3 Discussion

The first observation derived from the survey results is that there is an urgent demand for Ainu language dictionaries in the form of smartphone applications

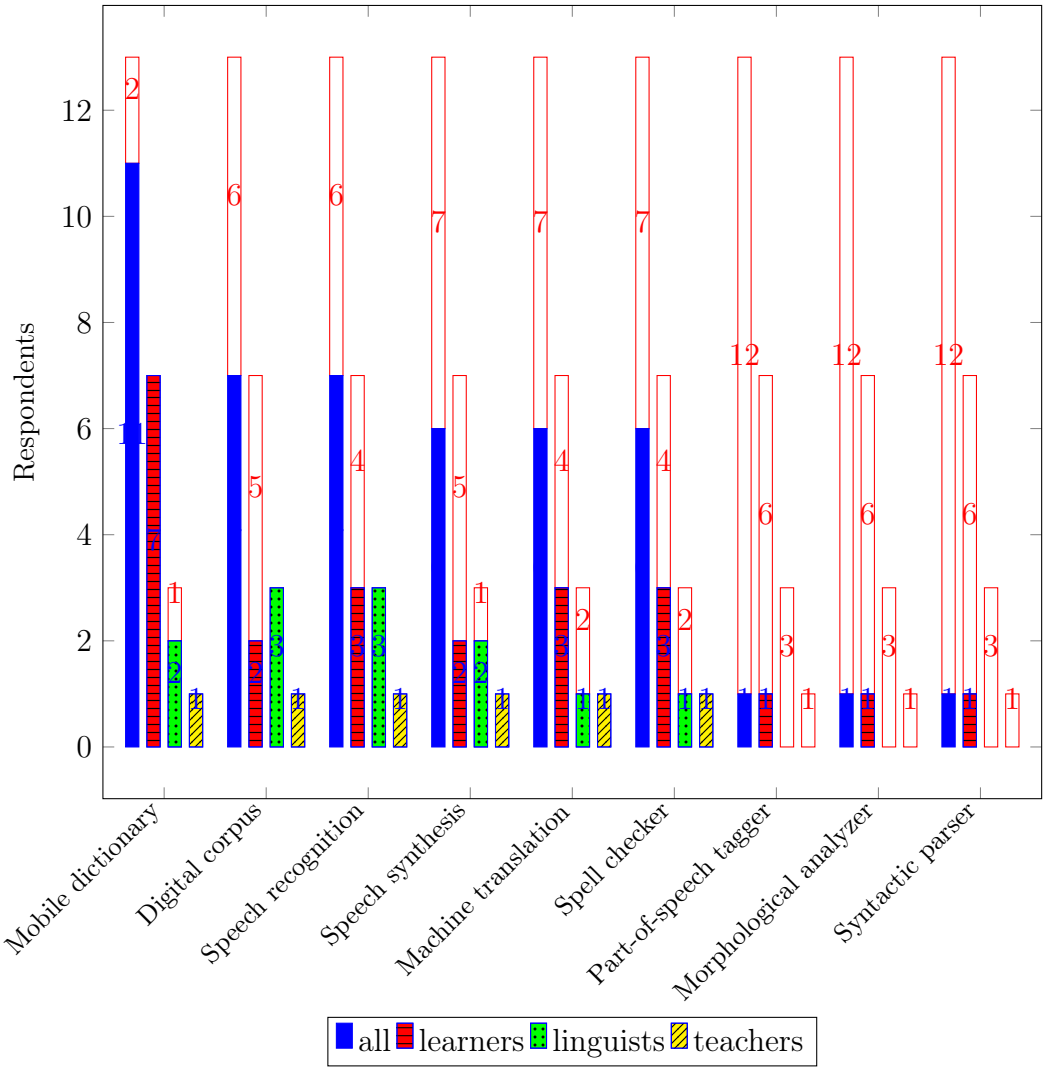

Figure 3.3: Respondents' expectations with respect to language-related technologies for the Ainu language (in general and by group). White bars represent the proportion of participants from the respective group who did not select the given answer.

(84.6% of respondents, including all the Ainu language learners, selected that option). While there exist very useful Web-based dictionaries, they are not optimized for use on mobile devices. Given the ubiquity of smartphones in today's world, mobile technology bears a great potential as a platform for the distribution of digital Ainu language resources and Ainu language-related technologies, which, however, is yet to be exploited.

Linguists, for obvious reasons, unanimously expressed the intent to work with an electronic corpus of Ainu, if one was available. Furthermore, they also seem to be particularly interested with speech recognition technology. This could be explained by the fact that the bulk of linguistic studies for Ainu involve manual transcription and analysis of speech data. Such work may be, at least to some extent, automated by utilizing a speech recognition system [81].

The goal of developing a large-scale corpus of Ainu, set out by Nowakowski, Ptaszynski and Masui [104], is justified not only by the fact that the majority of the participants support it. Apart from that, it will also play a central role in the development of other technologies called for by a large share of the respondents, such as speech technologies and machine translation.

Specialized tools for automated linguistic analysis (i.e. part-of-speech taggers, morphological analyzers and syntactic parsers) seem to be receiving little attention from the participants of the survey. Presumably, it stems from the fact that they have no experience with such technologies (recall the answers to Question 4, summarized in Figure 3.2), or – in the case of learners – have no need to use them directly. Nevertheless, I still consider those technologies important, as they will support the development of higher-level NLP technologies, such as machine translation [68, 99]. On the top of that, the annotations produced with their help will facilitate corpus linguistics studies on the Ainu language [46].

One more important conclusion can be drawn from the answers to Question 3, concerning the participants' dialects of interest. At present, most publicly available digital Ainu language materials (i.e. online dictionaries and text collections) belong to the relatively well-studied dialects of southern Hokkaidō, and in particular the Saru dialect. The results of my survey emphasize the need to build similar resources also for the varieties spoken in other regions, such as the Kushiro area and Sakhalin.

3.2 Main Tasks and Future Directions

3.2.1 Data Collection and Digitization

An essential prerequisite for carrying out linguistic research – and even more so when developing Natural Language Processing technologies – is to obtain as much relevant data as possible. As a first step in my research, I started collecting high-quality data already available in digital format. I estimate that the total amount of such data does not yet exceed 1 million words.

In the next phase, which is currently in progress, I am expanding the database by digitizing printed materials such as books and dictionaries. Furthermore, in the near future I plan to start experiments with training handwritten text recognition models for texts preserved only in the form of handwritten manuscripts, and speech recognition models for transcribing speech data.

3.2.2 Digital Corpus of the Ainu Language

In order to facilitate efficient use of the collected data, I initiated a project to organize it into a uniform format of an electronic corpus (for details, see Chapter 4).

The corpus will have two major applications:

1. It will play a pivotal role in my future research, and potentially in Ainu-oriented NLP research in general, as a source of data for training and evaluation. In Chapter 5 (Sections 5.2 and 5.3) I report the results of two experiments carried out so far, where the corpus was utilized.

2. I strongly believe (and survey results confirm it) that a large-scale corpus will be extremely useful for researchers and instructors of the Ainu language, especially if I enrich it with linguistic annotations and equip it with advanced search functions.

The main differences between my concept of the Ainu corpus and the existing digital resources, are as follows:

- **Scope and scale:** in recent years, an increasing number of research projects use the Web as a platform for the presentation of previously unpublished

Ainu language materials [e.g., 145, 24, 95]. Such materials typically originate from a single documentation project, involving one or several informants. On the contrary, my aim is to build a corpus composed of as many language materials as possible, including previously published texts.

- **Uniformity:** owing to the fact that my corpus aggregates a wide range of Ainu language documents from multiple data sets in a consistent format, it allows for querying all of them at once, via a single interface (rather than performing a separate search in each of those data sets).

- **Linguistic annotations:** due to high labor intensity of manual annotation projects, only a small portion of Ainu language materials include linguistic information other than transcription and translation into a reference language (usually Japanese). Using the available, linguistically analyzed materials (not only annotated texts but also dictionaries) in combination with Machine Learning techniques, I will generate annotations covering all texts in the corpus. To that end, in this thesis I describe my experiments aimed at improving the accuracy of automatic part-of-speech tagging for Ainu (see Chapter 5). In the near future I will use the results of that study to label the corpus with morpho-syntactic tags.

- **Advanced search functions,** known from high-quality digital corpora released for other languages (e.g., the British National Corpus[3]), such as frequency lists, concordances, or filtering by part-of-speech tags.

One of the main tasks for the future will be to expand the corpus with new materials, with special focus on those which have passed into the public domain and are no longer covered by copyright restrictions with regard to redistribution. Also, given the fact that nearly all Ainu language data, while being critical for linguistic research, lexicography and language education, has little or no commercial value, I hope that in the future it will be possible to reach a consensus with the copyright holders, allowing for unrestricted use.

The ultimate goal is to collect a corpus of texts spanning multiple regional varieties of Ainu. Such a resource will be useful for comparative linguistic studies

[3] https://www.english-corpora.org/bnc/

across dialects. I will also employ it as the training data in the development of a dialect identification tool, which will help researchers studying historical Ainu texts of uncertain provenance [see 133, p. 165].

An interesting idea to explore in the future (studied for instance by Cooper et al. [30]) would be to crowd-source further digital texts from the users of the corpus[4].

3.2.3 Text Processing Tools

The primary function of the aforementioned digital corpus is to provide the data needed for the development of Natural Language Processing tools. The first group of such tools are the specialized tools for automated text processing. They can be further subdivided into two categories:

1. Tools for text pre-processing, such as:

   - **Tokenizer:** divides the text into basic meaningful units (referred to as *tokens*). In the case of Ainu, orthographic words are often too coarse-grained, and therefore an additional process of word identification is required.
   - **Transcription normalization tool:** as mentioned in Section 2.3, many written documents do not fully conform to contemporary conventions for transcribing the Ainu language. As a consequence, an additional step of text normalization is necessary to resolve the inconsistencies when processing such materials.Furthermore, two distinct scripts are used to transcribe documents in the Ainu language: Latin alphabet and the *katakana* syllabary. This issue is partially addressed by the conversion software mentioned in Question 5 of the survey (see Section 3.1).

2. Tools for linguistic analysis, including:

   - **Part-of-speech tagger:** part-of-speech information is one of the most common types of linguistic annotation in digital corpora, useful both in NLP [68, 99] and linguistic research [46].

[4] A number of scholars report performing corpus analyses on private data sets of manually digitized Ainu texts. I strongly believe that – where there are no copyright-related restrictions – the community would greatly benefit from making such materials publicly available.

- **Morphological analyzer:** particularly useful for text analysis and processing in morphologically rich languages. Ainu exhibits a fairly complex agglutinative morphology, with elements of polysynthesis (such as incorporation and concentration of various morphemes in the verbal complex).
- **Syntactic parser:** analyzes the syntactic structure of a sentence.
- **Dialect identification tool:** for the analysis of historical documents of unknown provenance.

The tools will serve two purposes: firstly, they will help in producing linguistic annotations for the corpus, which in turn will be used to improve their coverage and accuracy. Secondly, I will make them publicly available, so that Ainu language experts can employ them in their research to analyze texts in a faster and more thorough manner.

For tasks where there is very little or no training data available (e.g., syntactic parsing) or the existing data is as yet insufficient to achieve a satisfactory level of accuracy, I will experiment with weakly-supervised approaches, such as transferring linguistic knowledge from other languages[5], or produce the necessary amount of training examples in an Active Learning scheme [136].

For detailed descriptions of the tools developed so far, see Chapter 5.

3.2.4 Dialogue System and Other Higher-level NLP Technologies

One of the long-term goals of my research is to develop an Ainu language Voice Dialogue System based on Artificial Intelligence, capable of holding simple conversations in Ainu. I believe that it will support the efforts of Ainu language learners and instructors by increasing the opportunities for learning through interaction (conversation), which is a crucial element of the language learning process [72, 58].

As a prerequisite, the Dialogue System will require the development of the following technologies, which by themselves will be of high importance:

[5]It must be noted, however, that such methods tend to yield the best results for closely related languages [see, e.g., 49], which is not a feasible scenario in the case of Ainu, as it has no known cognates. On the other hand, the fact that most texts are accompanied by parallel Japanese translations, makes it tempting to explore the said approaches.

- **Speech Recognition:** apart from voice-based interaction with the Dialogue System, it will also allow for automatic evaluation of the learners' pronunciation. Outside of the educational context, it will support the work of researchers transcribing speech data.

- **Speech Synthesis**: it will offer an easy way to create new audio content in the Ainu language, such as audiobooks for visually impaired. Separate models might be trained on the recordings of Ainu songs and *yukar*, narrative epics performed as metered chants, and then used to reconstruct the hypothetical manner in which the texts only preserved in writing would be performed.

- **Machine Translation** (between Ainu and Japanese, and possibly also between Ainu and English).

Conversational data in the Ainu language is extremely limited. A possible solution to that problem is to transfer conversational knowledge from resource-rich languages, such as Japanese. To achieve that, I will train a Japanese–Ainu Machine Translation (MT) system, based on the parallel corpus described in Chapter 4, and use it to translate the existing conversational data in Japanese (e.g. the Nagoya University Conversation Corpus [40]) to the Ainu language. The automatically translated data will then be used to pre-train the dialogue system [see 130], and finally I will perform fine-tuning with the few existing Ainu language conversational data sets, such as the materials for the "Ainugo Rajio Kōza" (see Section 2.2.3) and the textbook *First Step for the Sakhalin Ainu Language*, by Murasaki [90].

The final product will be released in the form of a Web platform and/or mobile application. It could also take the shape of a robot, which has the additional benefit of allowing for physical interaction. Afterwards, I will continue to improve it based on the input from users. In addition to the conversational system itself, I will also distribute its "by-products": the Speech Recognition, Speech Synthesis and Machine Translation technologies, as stand-alone systems.

As a first step towards the goal presented above, in Section 5.4 I describe a preliminary study to create a rule-based conversational system, using the SoftBank Pepper robot.

3.3 Conclusions

I introduced a framework for the research on the Ainu language utilizing modern computational techniques, along with ideas for future directions.

As an attempt to empirically examine the actual needs of the potential users of the technologies and resources created in this project, I conducted a survey study among a group of people participating in Ainu language-related activities. The results revealed a strong demand for digital resources and Natural Language Processing technologies for the Ainu language, such as mobile dictionaries, digital corpora and speech technologies.

As shown in Section 2.4, there have been a few studies – apart from my research – in the field of Natural Language Processing and Computational Linguistics, dedicated to the Ainu language. However, to my best knowledge, I am the first to take a holistic approach to the problem of providing the Ainu language community with access to language-related technologies, and to establish a long-term, empirically-based strategy for their development.

Chapter 4

Large-scale Digital Corpus of the Ainu Language

Rapid development of language technologies that we have experienced in last decades, would be impossible without the existence of large-scale digital language corpora. Apart from being crucial for the field of NLP, they are also powerful resources for linguistic studies. However, such resources are not available for the majority of 7000 existing languages[1], including Ainu. Due to its unique characteristics (such as noun incorporation or the usage of affixes – instead of pronouns – to express grammatical person), the Ainu language has been the subject of a number of linguistic and anthropolinguistic studies. However, most researchers, even if they apply modern digital technologies, only work on small amounts of language data. To initiate further development of Ainu language studies, I propose a unified corpus of the Ainu language. The corpus covers a wide range of existing documents, in a consistent structure that will enable large-scale linguistic analysis. It will also support the development of language technologies for the Ainu language, contributing to the ongoing process of Ainu language revitalization.

Large portions of this chapter have appeared in [101].

[1] According to Ethnologue [140], there are 7,097 living languages.

4.1 Background and Related Work

For English and other major languages, digital corpora have been compiled since the 1960s, one of the pioneering works in the field being the Brown Corpus [69]. As of today, some of the largest corpora available for these languages include hundreds of millions or even billions of words (e.g. [31, 123]). In recent years, corpora have been developed for some of the under-resourced or endangered languages, as well (e.g. [78, 4, 47, 43, 3]). In the case of Ainu, there is only one research project whose name includes the word *corpus*: A Glossed Audio Corpus of Ainu Folklore [95], a collection of 23 folktales spanning 7 hours of speech (44,717 words). In addition to that, there exist several online repositories of texts in the Ainu language [21, 24, 98, 145]. I will describe them in detail in the next section. Apart from studies focused on a particular language (or a language pair, in the case of parallel corpora), there is also an ongoing initiative, started by Steven Abney and Steven Bird [2], to build a large-scale multi-lingual corpus in a consistent format allowing for automatic processing. The idea has been picked up by Emerson et al. [35], who released the SeedLing, a resource containing small amounts of data from 1,451 languages, including 15 sentences in Ainu.

4.2 Selection of Texts for the Corpus

Today, the Ainu language is not used as a language of everyday communication, nor as an official language in any institutions. As a result, the amount of available texts in Ainu (in particular, digitized texts) is limited. For that reason, rather than selecting a sample of documents for the corpus, I want to collect and include as many existing materials as possible. That being said, in order to maintain high linguistic quality of my resource, some sort of selection criteria need to be defined. A first intuition would be not to accept texts by non-native speakers, but it must be remembered that the Ainu people – including some of those acting as informants in language documentation projects – have not been using Ainu as their first language of communication for several decades, therefore the question of what level of Ainu language proficiency should be regarded as sufficient is not trivial. For instance, the *Ainu Taimuzu*, a magazine in Ainu existing since 1997, is published

by the Ainu Language Pen Club led by an ethnic Japanese, Satoshi Hamada. On the other hand, its value as a rare example of Ainu being used in the context of contemporary affairs is indisputable and in the future, I shall consider including its contents in the corpus. Moreover, as I explain in Section 4.3.1, each text in the corpus is accompanied by metadata including detailed information about its authors – using that information the users will be able to narrow down their search to texts that satisfy their requirements.

4.2.1 Materials Included So Far

Below I introduce the Ainu language resources included in the corpus so far. The majority of them are products of language documentation projects. Altogether, they comprise a total of 1.86 million characters (410 thousand tokens). Table 4.1 shows the statistics of all seven datasets.

1. *Ainu shin-yōshū* [29]
   A collection of 13 epics (*yukar*) compiled by Yukie Chiri. For my corpus, I used a version with modernized transcription, published by Hideo Kirikae [65].

2. A Talking Dictionary of Ainu: A New Version of Kanazawa's Ainu Conversational dictionary [21]
   An online dictionary based on the *Ainugo kaiwa jiten* [55], a lexicon compiled by Shōzaburō Kanazawa and Kotora Jinbō, and published in 1898. It contains 3,847 entries, each of them consisting of a single word, multiple related words, a phrase or a sentence. For my corpus, I used the modernized transcription provided by Bugaeva et al. [21]. Apart from the Ainu text and Japanese and English translations, the dictionary contains information about morphology as well as part-of-speech annotations.

3. Glossed Audio Corpus of Ainu Folklore [95]
   A digital collection of 23 Ainu folktales narrated by Kimi Kimura, with glosses (morphological annotation) and translations into Japanese and English. At present, my corpus only includes the first 10 stories which had been released

at the time of collecting materials for it (December 2017). They comprise a total of 22,559 tokens of text in Ainu.

4. Dictionary of Ainu place names [87]
   A dictionary of Ainu place names in the form of a database. It contains a total of 3,152 topological names, along with an analysis of their components (including part-of-speech annotation) and Japanese translations.

5. Dictionary of the Mukawa dialect of Ainu [24]
   An online resource developed on the basis of 150 hours of speech in the Ainu language (Mukawa dialect), recorded by Tatsumine Katayama between 1996 and 2002 with two native speakers: Seino Araida and Fuyuko Yoshimura. It contains 6,284 entries, comprising a total of 64,656 tokens of text in Ainu.

6. Collection of Ainu Oral Literature [98]
   A digital collection of 98 texts in Ainu: 53 prose tales (*uwepeker*), 29 mythic epics (*kamuy yukar*), 11 heroic epics (*yukar*) and 5 texts of other types[2].

7. Ainu Language Archive – Materials [145]
   An online archive of texts in Ainu, based upon voice and video recordings obtained from three native speakers: Matsuko Kawakami, Toshi Ueda and Tatsujirō Kuzuno.

Table 4.1: Statistics of the Ainu language materials included in the corpus.

Text Collection	Characters total (excluding spaces)	Tokens total
Ainu shin-yōshū (revised by Kirikae [65])	36,780	8,786
A Talking Dictionary of Ainu... [21]	54,655	12,978
Glossed Audio Corpus of Ainu Folklore [95]	86,214	22,559
Dictionary of Ainu place names [87]	26,872	9,246
Dictionary of the Mukawa dialect of Ainu [24]	324,022	64,656
Collection of Ainu Oral Literature [98]	479,461	106,257
Ainu Language Archive – Materials [145]	856,555	185,919
Total	1,865,559	410,401

[2]For detailed description (in Japanese) see: http://www.town.biratori.hokkaido.jp/biratori/nibutani/pdf/ainu_gaiyou.pdf

4.2.2 Materials to Include in the Future

In the near future, I plan to expand my corpus with the following materials available online.

1. Ainu Language Material Release Project [53]
   An online repository of voice recordings obtained from two native speakers of Ainu: Matsuko Kawakami and Suteno Orita, accompanied by transcriptions (in kana and Latin alphabet) and translations into Japanese, released by the Research Institute for Languages and Cultures of Asia and Africa, Tokyo University of Foreign Studies. It contains 17 documents (mostly *uwepeker* – prose tales), comprising a total of 39,398 tokens of text in Ainu.

2. Ainu textbooks by FRPAC[3]
   A series of 24 textbooks for eight different dialects of Ainu (seven dialects spoken in Hokkaidō and Sakhalin Ainu), published by the Foundation for Research and Promotion of Ainu Culture.

3. Ainu Language Radio Course textbooks[4]
   A series of textbooks for the "Ainu-go Rajio Kōza" ("Ainu Language Radio Course"), broadcasted by the STV Radio in Sapporo since 1998.

4. The New Testament
   Translation of the New Testament into Ainu by a British missionary, John Batchelor [25], who spent more than sixty years among the Ainu people [126]. Batchelor had received no linguistic training and his works related to the Ainu language have been criticized by some experts (e.g. by Chiri Mashiho [126]). However, due to the relatively large size of the resource, as well as the uniqueness of its contents, I will definitely consider adding it to the corpus.

Furthermore, I plan to digitize the following printed materials.

1. *Ku sukup oruspe* [141]
   Memoirs of a speaker of the Ishikari dialect of Ainu – Kura Sunasawa – written

[3] https://www.ff-ainu.or.jp/web/learn/language/dialect.html
[4] https://www.stv.jp/radio/ainugo/index.html

in Ainu and Japanese, and edited by Hideo Kirikae. It contains ca. 10,000 words of text in Ainu [139].

2. *Akor Itak* [51]
   The first standard textbook of the Ainu language, published in 1994 by the Hokkaidō Utari Association (now Hokkaidō Ainu Association).

4.2.3 Representativeness and Balance

An important problem in corpus design is the selection of a sample of texts that is representative for the language in question [11]. However, due to the fact that I intend to include as many texts as possible in the corpus, that problem is irrelevant for my project. Of course, it does not mean that the corpus constitutes a balanced representation of the Ainu language – most existing documents are transcriptions of oral literature (such texts amount for nearly 80% of the materials already added to the corpus, listed above).

4.3 Elements and Structure of the Corpus

4.3.1 Metadata

The following information about each document is stored in a separate file ("`[document ID]-header.xml`"):

- Collection name (e.g. "Ainu shin-yōshū");
- Document title, if available;
- Author(s) of the text (i.e. informants), if known;
- Author(s) of transcription;
- Author(s) of translations, if available;
- Copyright information;
- Year of creation, if available;

- Year of publication, if available;
- Date of obtaining;
- Source of the material (bibliographic reference or URL);
- Type of language. Available options: "literary" (i.e. oral literature, prayers), "conversational", "other", "undefined";
- Genre, e.g. "yukar" (heroic epics), "kamuy yukar" (mythic epics), "menoko yukar" (women's epics), "uwepeker" (prose tales), "upaskuma" (teachings of the ancestors), "isoytak" (personal narrative), "yaysama" (improvised songs), "upopo" (sitting songs), "inonno itak" (prayers), "daily conversation", "memoirs", "press", "blog", "other", "undefined";
- Dialect of Ainu, if known.

In addition, there are separate metadata files for text collections (used for storing general information about each collection and copyright information) and for authors of texts (Ainu speakers acting as informants), explaining from which region each of them came and describing their backgrounds.

4.3.2 Parallel Text

The base type of information included in each document is the text in Ainu, transcribed in Latin alphabet. With the exception of a single resource ([87]) and some fragments of documents from other collections, document-level translations into Japanese are available. That allowed me to create an Ainu–Japanese parallel corpus, rather than just a monolingual corpus of Ainu. Moreover, English translations are present in [95] and [21], and Peterson [113] released translations of the *Ainu shin-yōshū* – I also included these materials in my parallel corpus.

The majority of texts in the corpus are transcripts of oral literature, which was traditionally recited in verses, not necessarily corresponding to sentences. Verse boundaries are reflected (by line breaks) both in transcription of the original speech and in Japanese translations. Knowing that the process of sentence alignment would be time- and labor-consuming, I decided to use that inherent structure of

the source material, and established verse as the unit of bitext alignment in those documents. Parallel text in this format is stored in XML files sharing the common suffix "`-verses`" in their file names. An example is shown in Listing 4.1. On the other hand, the text (and its Japanese and English translations) in [95] and [21] is generally split into sentences, therefore I store it in files with the suffix "`-sentences`".

Listing 4.1: Fragment from the parallel corpus, aligned at the level of verses. Collection: *Ainu shin-yōshū*. File: "`SYOS12-verses.xml`".

```
<verse id="v3">
 <text lang="ain">sine an to ta pet esoro sinot=as kor</text>
 <text lang="jpn">ある日川に沿って遊びながら</text>
</verse>
```

4.3.3 Annotations

For materials, where linguistic annotations are available, I created additional files, sharing the common suffix "-annotations". At present, the corpus contains the following types of annotation:

- **Part-of-speech (POS) annotation** – available for [21], [87] and a subset (5 out of 13 documents) of [29]. An example of part-of-speech annotated document is shown in Listing 4.2.
- **Word-level translation (into Japanese)** – available for [87], a subset (5 documents) of [29] and a subset (25 documents) of [145].
- **Morpheme-level glosses** – available for [95], [21] and a subset (18 documents) of [145]. An example is shown in Listing 4.3.

The structure of my corpus as described above was inspired by the data format proposed by Steven Abney and Steven Bird [1]. However, it was designed as a tentative model and in the future I shall consider converting the corpus into one of the standard formats used in corpus building, such as XCES[5], TEI[6] or FoLiA [150].

[5] http://www.xces.org/
[6] http://www.tei-c.org/

Listing 4.2: A fragment of part-of-speech annotated text. Collection: *Ainu shin-yōshū*. File: "SYOS12-annotations.xml".

```
<verse id="v3">
 <token id="t13">
  <text lang="ain">sine</text>
  <pos lang="jpn">連体詞</pos>
 </token>
 <token id="t14">
  <text lang="ain">an</text>
  <pos lang="jpn">自動詞</pos>
 </token>
 <token id="t15">
  <text lang="ain">to</text>
  <pos lang="jpn">名詞</pos>
 </token>
```

Listing 4.3: A fragment of text annotated with word-level translations and morpheme-level glosses. Collection: Ainu Language Archive – Materials. File: "AASI20-annotations.xml".

```
<verse id="v16">
 <token id="t87">
  <text lang="ain">rapokke</text>
  <tr lang="jpn">そのうちに</tr>
  <morph id="m98">
   <text lang="ain">rapok</text>
   <gloss lang="jpn">間</gloss>
  </morph>
  <morph id="m99">
   <text lang="ain">ke</text>
   <gloss lang="jpn">〜の</gloss>
  </morph>
 </token>
 <token id="t88">
  <text lang="ain">a=</text>
  <tr lang="jpn">（私の･）</tr>
  <morph id="m100">
   <text lang="ain">a=</text>
   <gloss lang="jpn">4.(他主)=</gloss>
  </morph>
 </token>
```

Chapter 5

Development of Natural Language Processing Technologies for the Ainu Language

This chapter is devoted to the Natural Language Processing tools developed in the course of my research. Firstly, in Section 5.1 I discuss the improvements introduced to POST-AL, a lexicon-based NLP toolkit originally developed by Ptaszynski and Momouchi [119]. After that, I report the results of applying corpus-driven techniques (leveraging the Ainu language corpus presented in Chapter 4) to the tasks of word segmentation (Section 5.2) and part-of-speech tagging (Section 5.3). In the remaining part of this chapter, I describe a preliminary experiment in developing an Ainu language-speaking robot, intended for use in language education.

Large portions of this chapter have appeared in [107, 103, 105].

5.1 Improving POST-AL, the Toolkit for Ainu Language Processing

5.1.1 Original System

In 2012, Ptaszynski and Momouchi [119] proposed POST-AL (Part-Of-Speech Tagger for the Ainu Language), a tool for computer-aided processing of the Ainu language. The toolkit performs five tasks:

- **Transcription normalization**: modification of parts of text that do not conform to modern rules of transcription (e.g., *kamui→kamuy*);
- **Word segmentation (tokenization)**: a process in which the text is divided into basic meaningful units (referred to as *tokens*). For writing systems using explicit word delimiters, tokenization is relatively simple. However, in some languages (such as Chinese) word boundaries are not indicated in the surface form, or orthographic words are too coarse-grained and need to be further analyzed – which is the case for many texts written in Ainu;
- **Part-of-speech tagging**: assigning a part-of-speech marker to each token;
- **Morphological analysis** (see Ptaszynski et al. [121]);
- **Word-to-word translation** (into Japanese).

One of the core elements of POST-AL is its dictionary base. Originally, it contained one dictionary, namely the *Ainu shin-yōshū jiten* by Kirikae [65]. In 2016 Ptaszynski, Nowakowski, Momouchi and Masui investigated the possibilities of improving the part-of-speech tagging function of the system by testing it with four different dictionaries. In this section I present the results of further tests comparing two wide-coverage dictionaries of the Ainu language: the *Ainu shin-yōshū jiten* and the A Talking Dictionary of Ainu: A New Version of Kanazawa's Ainu Conversational dictionary by Bugaeva et al. [21]. Moreover, I describe an attempt to combine both dictionaries into one database, in order to improve the system's overall performance.

Until recently there were no commonly accepted rules for transcribing the Ainu language (see Section 2.2.2). At the same time, due to the language shift towards

Japanese and its diminishing effect on the Ainu language competence, texts collected in earlier years remain important records of the language. The modernization of such texts involves two tasks:

- Modifying character representations of certain sounds, such as 'ch'→'c', 'sh'→'s', 'ui'→'uy', 'au'→'aw';
- Correcting word segmentation. Authors of older transcriptions tended to use less word delimiters (texts were divided according to poetic recitation rules, into chunks often containing multiple syntactic words). In the context of this research, it leads to an increase in the proportion of forms not covered in dictionaries. Thus, it is necessary to split some of the orthographic words into smaller units.

In order to facilitate the analysis and processing of such documents, Ptaszynski and Momouchi [119] developed a maximum matching algorithm-based tokenizer, including several heuristic rules for normalization of transcription in older texts. Ptaszynski et al. [117] investigated the possibility of adapting the tokenizer algorithm included in the Natural Language Toolkit[1] for segmenting Ainu, but it did not perform as well as the dedicated tokenizer. In the present research I improved the dictionary lookup algorithm applied in the tokenizer and enhanced the normalization rules.

Another functionality of POST-AL which I aimed to improve in the research presented here, is part-of-speech tagging. To achieve that, I modified the tagging algorithm by including the information about term frequency in the process of part-of-speech disambiguation.

5.1.2 Description of the Improved System

In this section, I present the technical details of modified algorithms for transcription normalization, word segmentation, and part-of-speech tagging.

[1] http://www.nltk.org/

5.1.2.1 Transcription Normalization Algorithm

The role of this preprocessing step is to detect all substrings of a given input string that are equal to the upper part of any of the transcription change rules shown in Table 5.1, and generate a list of all possible transcriptions, where each of such substrings is either substituted with its modern equivalent (the lower part of the corresponding rule) or retained without modification. Given an input string with n substrings to be potentially modified, a list of 2^n strings will be generated. An example is shown in Table 5.2. Unlike in the original version of POST-AL, transcription change rules are optional – the decision as to which of them should be applied (which of the possible permutations of the input string to select) is made in the next step by the word segmentation algorithm.

Table 5.1: Transcription change rules applied in part-of-speech tagger for the Ainu language (POST-AL).

Original Transcription														
ch	sh(i)	ai	ui	ei	oi	au	iu	eu	ou	mb	mp	b	g	d
c	s	ay	uy	ey	oy	aw	iw	ew	ow	np	np	p	k	t
Modern Transcription														

Table 5.2: A fragment of text before and after processing with the normalization algorithm.

Input String	**List of Output Strings**	**Meaning**
chepshuttuye	cepsuttuye cepshuttuye chepsuttuye chepshuttuye	"to exterminate fish"

5.1.2.2 Tokenization Algorithm

Input The type of input for the tokenizer depends on whether one wants to apply transcription normalization or not:

- **With transcription normalization**: if the text has to be corrected in terms of transcription, then the word segmentation algorithm takes a list of all possible transcriptions of a given input string, which has been generated in the previous stage;
- **Without transcription normalization**: if there is no need for transcription normalization, the input only includes one string (the one that has been provided by the user).

Word Segmentation Process Instead of applying a maximum matching algorithm, the new tokenizer performs a dictionary lookup in order to find a single token or the shortest possible sequence of tokens from the lexicon, such that after concatenation is equal to the input token. If the current processing pipeline includes transcription normalization, the tokenizer iterates through all variants of the input string and selects the one that allows for a complete match with the shortest sequence of lexicon items (an example is shown in Table 5.3). If more than one variant can be matched with a sequence containing a certain number of tokens, priority is given to the variants following the modern transcription rules listed in the lower part of Table 5.1 (the matching algorithm iterates through the strings with modernized transcription, before proceeding to the ones where change rules were not applied).

Table 5.3: A fragment of text before and after processing with the tokenization algorithm.

Input	Shortest Sequence of Tokens to Match
cepsuttuye cepshuttuye chepsuttuye chepshuttuye	cep sut tuye

There are two reasons why the transcription normalization process is finalized at the stage of word segmentation: firstly, in the two dictionaries applied as the training data in this research (see Section 5.1.3) there are more than one

hundred items containing character sequences included in Table 5.1 as obsolete transcription rules, which means that there are exceptions to those rules. Secondly, older texts often include long space-delimited segments which need to be split into multiple tokens by the word segmentation algorithm. As a result, character combinations corresponding to one of the transcription change rules often occur at token boundaries (an example is shown in Table 5.4), in which case no modification should be applied. That, however, becomes clear only after word segmentation has been performed. This means that the dictionary lookup algorithm described here not only detects word boundaries but also performs disambiguation of transcription change rules.

Table 5.4: Example of a situation where the transcription change rules should not be applied.

Input Token	**Strings Generated by the Transcription Normalization Algorithm**	**Gold Standard Transcription and Word Segmentation**	**Meaning**
setautar	setawtar setautar	seta utar	"dogs"

Code implementing the transcription normalization and word segmentation algorithms described above is included in Appendix B.

5.1.2.3 Part-of-Speech Tagger

Originally, POST-AL performed part-of-speech disambiguation based on usage examples included in the dictionary. Namely, for each ambiguous token found in the input text it extracted lexical n-grams containing that token, searched for them in the dictionary and returned the part-of-speech tag of the candidate entry with the highest number of matches. However, the training data extracted from lexicons is very sparse, which means that for many n-grams there exist few or no relevant samples. To compensate for that, I created a modified tagging algorithm, which in such cases also takes into account the term frequency (number of occurrences in the training set) of each candidate term and returns the tag assigned to the item with the highest value. For instance, the form *sak* used as transitive verb (meaning 'to lack; not to have') appears 14 times in the dictionary, whereas the

noun *sak* ('summer') has three occurrences, which means that according to the proposed disambiguation method "transitive verb" should be selected as the POS tag for the token *sak*.

In order to verify the performance of different part-of-speech disambiguation methods, three variants of the tagging algorithm were tested:

- With n-gram based POS disambiguation (as in the original POST-AL system);
- With TF (term frequency) based POS disambiguation;
- N-grams + TF (TF based disambiguation is only applied to cases where n-gram based disambiguation is insufficient). Pseudocode of this variant is presented in Algorithm 5.1.

Algorithm 5.1 Part-of-speech disambiguation method based on lexical n-grams and term frequency

$W \leftarrow$ sequence of words to label: $(w_1, w_2, ..., w_n)$
$D \leftarrow$ dictionary, where each entry d consists of headword (h), part-of-speech tag (t), and list of usage examples (E)
for each $w_i \in W$ **do**
  $G \leftarrow$ n-grams (2-grams) including w_i: (w_{i-1}, w_i), (w_i, w_{i+1})
  // Find dictionary entries where the headword matches the target word
  $D' \leftarrow \{(h, t, E) \in D \mid h = w_i\}$
  // Find the entry (or entries) with the highest number of examples including n-grams from G
  $D' \leftarrow \arg\max_{(h,t,E) \in D'} |\{e \in E \mid \exists g \in G : g \text{ is substring of } e\}|$
  // Filter candidate entries by term frequency (i.e., the number of examples they include)
  $D' \leftarrow \arg\max_{(h,t,E) \in D'} |E|$
  label w_i with the tag t of the first entry in D'
end for

5.1.3 Dictionaries

5.1.3.1 Ainu shin-yōshū jiten

Initially, Ptaszynski and Momouchi [119] trained their system using the information extracted from a single dictionary, namely the *Ainu shin-yōshū jiten* by Kirikae [65]. It is a lexicon to Yukie Chiri's *Ainu shin-yōshū* ("Ainu Songs of Gods", a collection of 13 *yukar* stories first published in 1923). The dictionary contains 2,019 entries. Each headword is assigned with a part-of-speech tag. Additionally, a subset of the entries include usage examples. Later I will refer to this dictionary as "KK". Listing 5.1 shows a sample entry.

Listing 5.1: An entry from the *Ainu shin-yōshū jiten* by Kirikae, converted to XML format (KK dictionary).

```
<word>aep</word>
<morph>a{2}-e{1}-p{1}</morph>
<pos>名詞 [noun]</pos>
<tr>食べ物 [food]</tr>
<ref>aep'omuken</ref>
```

5.1.3.2 Ainu Conversational Dictionary

The *Ainugo kaiwa jiten* ("Ainu conversational dictionary") [55] was one of the first dictionaries of the Ainu language, compiled by a Japanese researcher, Shōzaburō Kanazawa, who visited Hokkaidō several times between 1895 and 1897. He published the dictionary in 1898 under the supervision of professor Kotora Jinbō. The original dictionary contains 3,847 entries, most of which presumably belong to the Saru dialect [19].

In 2010, Bugaeva et al. [21] released the A Talking Dictionary of Ainu: A New Version of Kanazawa's Ainu Conversational Dictionary, which is an online dictionary of Ainu, based on the original *Ainugo kaiwa jiten*. Apart from the original content (Ainu words and phrases and their Japanese translations), the dictionary provides additional information, including corrected Latin transcription, modern Japanese translations, part-of-speech classification, and English translations. Furthermore, with the help of a native speaker of Ainu (Setsu Kurokawa) mistakes and misinter-

Listing 5.2: An entry from A Talking Dictionary of Ainu: A New Version of Kanazawa's Ainu Conversational Dictionary.

```
此村に何か食物があるか  [Japanese translation as in the original lexicon [55]]
Tan kotan ta nepka aep an ruwe he an?  [original Latin transcription [55]]
tan kotan ta nep ka aep an ruwe an?  [modernized Latin transcription]
タン コタン タ ネプ カ アエプ アン ルウェ アン?  [transcription in kana syllabary]
この 村 に 何 か 食べ物 ある こと ある  [word-to-word translation to Japanese]
【連体】【名】【格助】【疑問】【副助】【名】【自】【形名】【自】  [part-of-speech annotation]
dem n pp n.interr adv.prt n vi nmlz vi  [part-of-speech annotation]
この村に何か食べ物はありますか？  [modern Japanese translation]
Is there anything to eat in this village?  [English translation]
tan kotan ta nep ka a-e-p an ruwe an  [morphemes]
this village at what even INDF.A-eat-thing exist.SG INFR.EV exist.SG  [gloss]
```

pretations found in the original dictionary were corrected [19]. A sample entry is presented in Listing 5.2.

In 2015, a revised version of the above-mentioned online dictionary has been released under the name of A Topical Dictionary of Conversational Ainu [96].

In my experiments, I used the A Talking Dictionary of Ainu: A New Version of Kanazawa's Ainu Conversational Dictionary. Apart from isolated headwords, the original dictionary included 2,459 multi-word items (phrases and sentences). Such entries were divided into separate single-word entries. The original multi-word entries were added to the modified dictionary as usage examples, which can be used by the part-of-speech tagging system for disambiguation. Finally, I performed automatic unification of duplicate entries (entries containing words appearing in multiple entries of the original dictionary) was performed, using the translations provided Bugaeva et al. to determine which headwords to merge, and which should be retained as separate homonymous entries. The basic format of the entries has been adjusted to conform to the dictionary format required by POST-AL (the same format which is also used in the *Ainu shin-yōshū jiten*). Each entry contains the following information: headword, morphological boundaries, part(s) of speech (in Japanese and English), Japanese and English translation and morpheme-to-morpheme interpretation (explanations of meaning or function of each morpheme). Furthermore, 1,496 entries contain usage examples with Japanese and English translations. The last modification I performed was excluding 62 usage examples (428 words) from the dictionary, in order to use them as test data in evaluation experiments (see Section 5.1.4). Total number of usage examples in the final version

of the dictionary is 12,513 (including duplicates). In the following sections I will refer to this dictionary as "JK". A fragment is presented in Listing 5.3.

Listing 5.3: An entry from the JK dictionary.

```
<word>aep</word><kana>アエプ</kana>
<morph>a-e-p</morph><pos>名詞</pos>
<pos_en>n</pos_en>
<tr>食べ物</tr><tr_en>food</tr_en>
<ex>tan kotan ta nep ka aep an ruwe an?</ex>
<ex_jp>この村に何か食べ物はありますか？</ex_jp>
<ex_en>Is there anything to eat in this village?</ex_en>
<ge>INDF.A-eat-thing</ge>
```

5.1.3.3 Combined Dictionary

In order to increase the system's versatility, I decided to combine the two dictionaries described in Sections 5.1.3.1 and 5.1.3.2 into one dictionary base. To achieve this, I extracted entries containing items listed in both dictionaries and automatically unified them, based on their Japanese translations (namely, homonymous entries with at least one *kanji* character in common have been unified and the rest was retained as separate entries). That resulted in a dictionary containing 4,161 entries. In the following sections I will refer to this dictionary as "JK+KK". Listing 5.4 shows an entry from this resource.

5.1.4 Test Data and Gold Standard

For evaluation of the proposed system I used four different datasets:

- **Yukar epics**: Five out of thirteen *yukar* stories (no. 9–13) from the *Ainu shin-yōshū* [29]. Apart from the original version by Chiri, I also used the variants edited by Kirikae, who manually corrected their transcription and word segmentation according to modern linguistic conventions, and included them in the *Ainu shin-yōshū jiten* [65]. The modernized version comprises a total of 1608 tokens. Later I refer to this dataset as "Y9–13". In the experiment with POS tagging, I only used a subset of the data in question, namely the story

Listing 5.4: An entry from the JK+KK dictionary.

```
<word>aep</word><kana>アエプ</kana>
<morph_kk>a$^{2}$-e$^{1}$-p$^{1}$</morph_kk>
<morph_jk>a-e-p</morph_jk>
<pos_jk>名詞</pos_jk>
<pos_kk>名詞</pos_kk>
<pos_en>n</pos_en>
<tr>食べ物</tr><tr_en>food</tr_en>
<ex>tan kotan ta nep ka aep an ruwe an?</ex>
<ex_jp>この村に何か食べ物はありますか？</ex_jp>
<ex_en>Is there anything to eat in this village?</ex_en>
<ge>INDF.A-eat-thing</ge>
<ref> aep'omuken</ref>
```

no. 10: *Pon Okikirmuy yayeyukar "kutnisa kutunkutun"* ["Kutnisa kutunkutun" – a song Pon Okikirmuy sang], which has 189 tokens – later it will be abbreviated to "Y10". A fragment is presented in Table 5.5.

- **Samples from the Ainu Conversational Dictionary**: Sixty two sentences (428 tokens) from the A Talking Dictionary of Ainu: A New Version of Kanazawa's Ainu Conversational Dictionary, which were excluded from the training data (see Section 5.1.3.2). Apart from the original text by Jinbō and Kanazawa [55], I also used the modernized version by Bugaeva et al. [21]. Later I refer to this dataset as "JK samples". A fragment is shown in Table 5.6.

- **Shibatani's colloquial text samples**: Both datasets mentioned above are either obtained directly from one of the dictionaries applied as the training data for the system (Ainu Conversational Dictionary) or from the compilation of *yukar* stories on which one of these dictionaries was based (*Ainu shin-yōshū*). To investigate the performance with texts unrelated to the system's training data, I decided to apply other datasets as well. As the first one I used a colloquial text sample included in *The Languages of Japan* [139], namely a fragment (154 tokens) of Kura Sunasawa's memoirs written in the Ainu language, *Ku sukup oruspe* ("My life story") [141], transcribed according to modern linguistic rules. Later I refer to this dataset as "Shib.";

- **Mukawa dialect samples**: I also used a sample (11 sentences, 87 tokens)

from the Japanese–Ainu Dictionary for the Mukawa Dialect of Ainu [24]. It is a transcribed version of audio materials Tatsumine Katayama recorded between 1996 and 2002 with two native speakers of the Mukawa dialect of Ainu: Seino Araida and Fuyuko Yoshimura, containing 6,284 entries. Later I refer to this dataset as "Muk."

Table 5.5: A fragment from the Y9–13; original text by Chiri (top) and postprocessed by Kirikae (middle).

Shineantota petetok un shinotash kushu payeash awa
Sine an to ta petetok un sinot as kusu paye as a wa
Meaning: "One day when I went for a trip up the river"

Table 5.6: A sentence from the JK samples; original version (top) and with transcription normalized by Bugaeva et al. (middle).

Tambe makanak an chiki pirika?
Tanpe makanak an ciki pirka?
Meaning: "What should I do about it?"

Below I describe the variants of the test data applied in testing each element of the system.

5.1.4.1 Test Data for Transcription Normalization

To test the transcription normalization performance I used two datasets: Y9–13 and JK samples. Each of them was prepared in two versions:

- **Original ("O")**: Original texts by Chiri or Jinbō and Kanazawa, without any modifications;
- **Original, with spaces removed ("O-SR")**: Original texts by Chiri and Jinbō and Kanazawa, preprocessed by removing any word segmentation (whitespaces) from each line.

As the gold standard data, I used the modernized versions of both texts [65, 21].

5.1.4.2 Test Data for Tokenization

For evaluation of the tokenizer, I prepared two different versions of the test data:

- **Modern transcription, spaces removed ("M-SR")**: The first variant includes all four datasets. In the case of Y9–13 and JK samples, modernized versions by Kirikae and Bugaeva et al. were used. Each line of text was preprocessed by removing whitespaces;
- **Original spaces, modernized transcription ("O/M")**: In this variant, only two datasets were used: the Y9–13 and JK samples. I retained the word segmentation (usage of whitespaces or lack thereof) of the original texts [29, 55]. However, in order to prevent differences between transcription rules applied in original and modernized texts from affecting word segmentation experiment results, the texts were preprocessed by unifying their transcription with the modern (gold standard) versions. Table 5.7 shows to what extent the word segmentation of original texts is consistent with modernized versions by Kirikae and Bugaeva et al. (the evaluation method is explained in Section 5.1.5).

Table 5.7: Results of the evaluation of word segmentation in original texts by Chiri or Jinbō and Kanazawa against modern versions.

	Y9–13	**JK Samples**	**Overall**
Precision	0.998	0.949	0.983
Recall	0.609	0.918	0.674
F-score	0.756	0.933	0.800

The reason for performing the experiment on two versions of the test data was to verify which approach is more effective: retaining whitespaces even if in some cases it may cause segmentation errors, or removing any word segmentation used in the original text. To illustrate the problem, below is a fragment of text from Y9–13 in original transcription by Chiri [29], the modern transcription by Kirikae [65], and two versions prepared for the experiment:

- **Original transcription**: unnukar awa kor wenpuri enantui ka;

- **Modern transcription (gold standard)**: un nukar a wa kor wen puri enan tuyka;
- **Modern transcription, spaces removed**: unnukarawakorwenpurienantuyka;
- **Modern transcription, original word segmentation**: unnukar awa kor wenpuri enantuy ka;
- **Meaning**: "When she found me, her face [took] the color of anger."

Modern transcriptions of all four texts [65, 21, 139, 24] were used as the gold standard.

5.1.4.3 Test Data for POS Tagging

To evaluate part-of-speech tagging performance, I used two texts: Y10 and JK samples, both of them in modern transcription [65, 21]. Gold standard POS annotations were obtained from professor Yoshio Momouchi (see Momouchi et al. [86]) and Bugaeva et al. [21], respectively.

Table 5.8 presents the statistics of all four datasets used for evaluation, including their different variants.

5.1.5 Evaluation Methods

All experimental results were calculated with the means of Precision (P), Recall (R), and balanced F-score (F_1). In this section I provide definitions of the evaluation metrics for each type of experiments and describe the evaluation methodologies.

5.1.5.1 Balanced F-Score

Balanced F-score is the harmonic mean of Precision and Recall and its calculation method is the same across all experiments:

$$F_1 = 2\frac{PR}{(P+R)}. \tag{5.1}$$

Table 5.8: Statistics of the samples used for testing.

Data		Variant	Characters (Excluding Spaces)	Tokens
Ainu shin-yōshū	**Y9–13**	O*	6,883	1,076
		O-SR**		N/A
		M-SR***	6,501	N/A
		O/M****		1,076
		Kirikae [65]	6,501	1,608
	Y10	Kirikae [65]	822	189
Ainugo Kaiwa Jiten/ A Talking Dictionary of Ainu... **(JK samples)**		O	1,742	418
		O-SR		N/A
		M-SR	1,617	N/A
		O/M		416
		Bugaeva et al. [21]	1,617	428
Shibatani's colloquial text samples **(Shib.)**		Shibatani [139]	583	154
Mukawa dialect samples **(Muk.)**		Chiba University... [24]	341	87

* Original transcription.

** Original transcription with spaces removed.

*** Modern transcription with spaces removed.

**** Modern transcription with original spaces.

5.1.5.2 Evaluation of Transcription Normalization

In the case of transcription normalization, Precision is calculated as the percentage of correct single-character edits (deletions, insertions or substitutions) within all edits performed by the system, and Recall as the percentage of correct edits performed by the system within all edits needed to normalize the transcription of a given text. The total number of edits performed is equal to the Levenshtein distance between the input and the output, and the total number of edits needed, to the Levenshtein distance between the input and the gold standard. To calculate the number of correct edits I used a combination of the Levenshtein distance between the input and the output, and between the output and the gold standard (for each edit performed by the system I checked if it reduced the edit distance to the gold standard).

$$P = \frac{correct\ \ edits}{all\ \ returned\ \ edits} \tag{5.2}$$

$$R = \frac{correct\ \ edits}{all\ \ gold\ \ standard\ \ edits}. \tag{5.3}$$

As was explained in Section 5.1.2.1, the process of transcription normalization is finalized at the stage of tokenization. Therefore, after processing all texts with the transcription normalization algorithm, they were also processed with the tokenizer. In order to prevent tokenization errors from affecting the evaluation results, whitespaces were removed from both the output texts and the gold standard texts.

5.1.5.3 Evaluation of Tokenization

In the case of tokenization, Precision is calculated as the proportion of correct separations (spaces) within all separations returned by the system, whereas Recall is the number of correct spaces the system returned divided by the number of spaces in the gold standard.

$$P = \frac{correctly\ \ predicted\ \ spaces}{all\ \ returned\ \ spaces} \tag{5.4}$$

$$R = \frac{correctly\ \ predicted\ \ spaces}{all\ \ gold\ \ standard\ \ spaces}. \tag{5.5}$$

5.1.5.4 Evaluation of Part-of-Speech Tagging

In this case, Precision is calculated as the percentage of correct annotations within all annotations made by the system. Recall is the percentage indicating how many correct annotations the system returned compared to the gold standard.

$$P = \frac{correct\ \ annotations}{all\ \ system's\ \ annotations} \tag{5.6}$$

$$R = \frac{correct\ \ annotations}{all\ \ gold\ \ standard\ \ annotations}. \tag{5.7}$$

The dictionary by Kirikae [65] and the A Talking Dictionary of Ainu differ in terms of part-of-speech classification. Furthermore, Momouchi et al. [86] annotated Y10 according to yet another part-of-speech classification guidelines, based on the analyses of Nakagawa [92], Tamura [143, 144], Kirikae [65] and Katayama [56]. Therefore, in order to evaluate the POS tagging experiment results, I used the part-of-speech conversion tables built in POST-AL (see [119]). Specifically, I converted all annotations to the part-of-speech classification standard used by

Nakagawa [92]. The conversion method is shown in Table 5.9. As an alternative method for the evaluation of tagging results, I used a simplified POS standard (Table 5.10), where subclasses of general word classes are not differentiated (e.g., intransitive and transitive verbs are both considered the same class: "verb"), thus in Section 5.1.6.3 two different results are given for each experiment, depending on the part-of-speech conversion table used ("Nakagawa" and "simplified").

5.1.6 Results and Discussion

5.1.6.1 Transcription Normalization

Table 5.11 shows the results of transcription normalization experiments. System trained on Kirikae's lexicon achieved the highest scores for the Y9–13 dataset, which is not surprising, since the dictionary is based on *yukar* epics. In the case of JK samples, however, performance with the combined dictionary (JK+KK) was as good as with the JK dictionary only. Furthermore, the combined dictionary achieved the best overall results. The results for texts with original word segmentation retained were slightly better in all test configurations. Relatively low values of Recall in experiments on JK samples, observed across all combinations of dictionaries and input text versions, can be explained by a high occurrence of forms transcribed according to non-standard rules modified by Bugaeva et al. in the modernized version of the dictionary, but not included in the list of transcription change rules applied in this research, such as 'ra'→'r' (e.g., *arapa*→*arpa*), 'ri'→'r' (e.g., *pirika*→*pirka*), 'ru'→'r' (e.g., *kuru*→*kur*), 'ro'→'r' (e.g., *koro*→*kor*) or 'ei'→'e' (e.g., *reihei*→*rehe*). This has two reasons: firstly, these rules are not universal (they are observed only in a portion of texts requiring normalization). Secondly, preliminary experiments revealed that employing these rules can negatively affect the performance of the system when processing texts to which they don't apply (e.g., the *Ainu shin-yōshū*).

5.1.6.2 Tokenization

The results of tokenization experiments are shown in Table 5.12. Table 5.13 shows a fragment from Y9–13 (M-SR) before and after segmentation. Similarly to transcription normalization, the tokenization algorithm also performed the best

Table 5.9: Table for conversion of other Ainu part-of-speech standards into Nakagawa's standard.

JK	KK	Momouchi et al. [86]		Nakagawa [92]
完全動詞 (complete verb)	ゼロ項動詞 (complete verb)	完全動詞 (complete verb)	→	0 項動詞 (complete verb)
自動詞 (intransitive verb)	一項動詞 (intransitive verb)	自動詞 (intransitive verb)	→	1 項動詞 (intransitive verb)
他動詞 (transitive verb)	二項動詞 (transitive verb)	単他動詞 (transitive verb)	→	2 項動詞 (transitive verb)
複他動詞 (ditransitive verb)	三項動詞 (ditransitive verb)	複他動詞 (ditransitive verb)	→	3 項動詞 (ditransitive verb)
	人称代名詞 (personal pronoun)	人称代名詞 (personal pronoun)	→	代名詞 (pronoun)
	指示代名詞 (demonstrative pronoun)		→	代名詞 (pronoun)
	疑問不定代名詞 (interrogative indefinite pronoun)	疑問代名詞 (interrogative pronoun)	→	疑問詞 (interrogative)
	疑問不定副詞 (interrogative indefinite adverb)	疑問副詞 (interrogative adverb)	→	疑問詞 (interrogative)
	指示副詞 (demonstrative adverb)		→	副詞 (adverb)
後置副詞 (postpositive adverb)	後置詞的副詞 (postpositive adverb)	後置副詞 (postpositive adverb)	→	副詞 (adverb)
	指示連体詞 (demonstrative prenoun adjectival)		→	連体詞 (prenoun adjectival)
	後置詞 (postposition)		→	格助詞 (case particle)
	名詞的助詞 (nominal particle)		→	名詞 (noun)

Table 5.10: POS conversion table, simplified standard.

Nakagawa [92]		**Simplified Standard**
0 項動詞 / 1 項動詞 / 2 項動詞 / 3 項動詞 (complete verb / intransitive verb / transitive verb / ditransitive verb)	→	動詞 (verb)
固有名詞 / 代名詞 / 位置名詞 / 形式名詞 (proper noun / pronoun / locative noun / expletive noun)	→	名詞 (noun)
格助詞 / 接続助詞 / 副助詞 / 終助詞 (case particle / conjunctive particle / adverbial particle / final particle)	→	助詞 (particle)
人称接辞 / 接頭辞 / 接尾辞 (personal affix / prefix / suffix)	→	接辞 (affix)

for *yukar* stories (Y9–13) when coupled with the *Ainu shin-yōshū jiten* (KK). Analogically, for JK samples, the JK dictionary was the best. It shows a weak point of the presented segmentation algorithm: while adding new forms to the lexicon improves its versatility (ability to process texts from different domains), it also increases the number of possible mistakes the tokenizer can make with texts for which the original lexicon had been (nearly) optimal.

The combined dictionary performed better than the other two dictionaries on test data unrelated to the training data (Shib. and Muk.), and also achieved the best overall results (F-score). On the other hand, overall Recall was higher with the KK dictionary. To some extent it can be explained by the differences in word segmentation between the two dictionaries: many expressions (e.g., *oro wa*, 'from' or *pet turasi*, 'to go upstream') written as two separate segments by Kirikae (both in the lexicon part of the *Ainu shin-yōshū jiten* and in his modernized transcriptions of the *yukar* epics), are transcribed as a single unit (*orowa*, *petturasi*) by Bugaeva et al. [21]. Once these forms are added to the lexicon, the word segmentation algorithm, which prefers long tokens over shorter ones, stops applying segmentation to the tokens *orowa* and *petturasi*, resulting in lower Recall. This phenomenon occurs in the opposite direction as well: the only two types of tokenization errors made in the JK samples (O/M) when the combined dictionary was used, but not with the JK dictionary, were both of this type – the expressions transcribed by Bugaeva et al. as *somo ki* ('do not') and *te ta* ('here') are listed as *somoki* and *teta* in the *Ainu shin-yōshū jiten*.

As discussed in Section 5.1.1, authors of older documents in the Ainu language

Table 5.11: Transcription normalization experiment results (best results in bold).

			Y9–13	JK Samples	Overall	Input Text Version:
DICTIONARY	JK	Precision	0.871	0.942	0.885	O-SR
		Recall	0.897	0.658	0.833	
		F-score	0.884	0.775	0.859	
		Precision	0.890	**0.956**	0.903	O
		Recall	0.897	**0.658**	0.833	
		F-score	0.893	**0.780**	0.867	
	KK	Precision	0.967	0.899	0.954	O-SR
		Recall	**0.966**	0.628	0.876	
		F-score	0.966	0.740	0.913	
		Precision	**0.980**	0.926	**0.969**	O
		Recall	0.958	0.628	0.871	
		F-score	**0.969**	0.749	0.917	
	JK+KK	Precision	0.953	0.942	0.951	O-SR
		Recall	0.964	0.658	**0.883**	
		F-score	0.958	0.775	0.916	
		Precision	0.971	**0.956**	0.968	O
		Recall	0.958	**0.658**	0.879	
		F-score	0.964	**0.780**	**0.921**	

tended to use spaces more sparingly than contemporary experts. The opposite (i.e., insertion of a whitespace where it would not be inserted in a modern transcription), however, is not frequent. In line with that observation, the tokenizer yielded higher scores for test data with original word boundaries retained (Y9–13 (O/M) and JK samples (O/M)) than for the variants where spaces were removed. This means that the original word segmentation contributes positively to the tokenization process.

Table 5.13: A fragment from Y9–13 (M-SR) before and after tokenization.

Input:	kekehetakcepsuttuyecikikusnena
Tokenizer output:	keke hetak cep sut tuye ciki kusne na
Gold standard:	keke hetak cep sut tuye ci ki kusne na
Meaning:	“Now I’m going to show you how to make fish extinct” [113]

Table 5.12: Tokenization experiment results (best results in bold).

DICTIONARY			Y9–13	JK Samples	Y9–13 + JK Samples	Shib. + Muk.	Overall	Input Text Version:
DICTIONARY	JK	Precision	0.575	**0.935**	0.634	0.742	0.644	M-SR
		Recall	0.772	0.907	0.801	0.808	0.801	
		F-score	0.659	0.921	0.708	0.774	0.714	
		Precision	0.652	0.933	0.700	n/a	n/a	O/M
		Recall	0.894	**0.984**	0.913	n/a	n/a	
		F-score	0.754	**0.957**	0.792	n/a	n/a	
	KK	Precision	0.921	0.703	0.867	0.649	0.838	M-SR
		Recall	0.889	0.842	0.879	0.822	**0.873**	
		F-score	0.905	0.766	0.873	0.726	0.855	
		Precision	**0.950**	0.772	0.904	n/a	n/a	O/M
		Recall	**0.944**	0.981	**0.952**	n/a	n/a	
		F-score	**0.947**	0.864	0.928	n/a	n/a	
	JK+KK	Precision	0.905	0.943	0.913	**0.776**	**0.896**	M-SR
		Recall	0.854	0.896	0.863	**0.860**	0.863	
		F-score	0.879	0.919	0.887	**0.816**	**0.879**	
		Precision	0.939	0.932	**0.937**	n/a	n/a	O/M
		Recall	0.919	0.975	0.931	n/a	n/a	
		F-score	0.929	0.953	**0.934**	n/a	n/a	

5.1.6.3 Part-of-Speech Tagging

The results of part-of-speech tagging experiments are presented in Table 5.14. Table 5.15 shows a fragment from the JK dictionary analyzed with POST-AL (with both POS annotations and word-to-word translation into Japanese). The results indicate that as long as the tagger is trained with language data belonging to the same type as the test data (i.e., classical Ainu of the *yukar* epics, also covered in the KK dictionary, and colloquial language of the JK dictionary), part-of-speech disambiguation based on lexical n-grams is more accurate than the method using Term Frequency. But it also shows that combining both approaches (with priority given to n-grams) provides the best performance in each case. While to a certain extent the most frequent tag approach compensates for the shortcomings of lexical context-based disambiguation method in low-data conditions, it is far from perfect. Firstly, it ignores the context in which the given token appears, and secondly, in this case the frequency of each tag is calculated from usage examples included in

the dictionary, which by no means can be regarded as a balanced representation of the language. As a result, the tagger still makes a considerable amount of disambiguation errors (see Table 5.16).

I also found that although the JK dictionary and KK dictionary belong to different domains (colloquial and classical language), combining them both improved overall POS tagging performance and in the case of the Y10 dataset yielded the best results of all combinations.

There is a gap between the results of tagging Y10 and JK samples, which can be partially explained by differences in part-of-speech classification of certain items between the two dictionaries applied in the system and the gold standard annotations produced by Momouchi et al. [86]. For example, Momouchi et al. annotated the copula *ne* as "auxiliary verb", whereas in the training data it is listed as "transitive verb". In the experiment with Y10, the JK+KK dictionary, and the n-gram+TF based tagging algorithm (the combination that yielded the best result for Y10), this token and other errors of this type accounted for 61% of all incorrect predictions (see Table 5.16).

Table 5.16: Statistics of POS tagging errors in the experiment with Y10, the JK+KK dictionary, and the n-gram+TF based tagger.

Type of Error	**Count**
Tagger (disambiguation error)	8 (35%)
Dictionary (out-of-vocabulary item)	1 (4%)
POS classification (the same word, but different tag)	14 (61%)

5.1.7 Conclusions

In this section I presented my research in improving POST-AL, a tool for computer-aided processing of the Ainu language. In addition to improving the algorithms for transcription normalization, word segmentation, and part-of-speech tagging, I also expanded the system's dictionary base by combining two comprehensive Ainu language dictionaries. I found out that the combination improved overall performance of the tools, especially with objective samples unrelated to the training data.

Table 5.14: Part-of-speech tagging experiment results (best results in bold).

			Test Data						Tagging Algorithm Version:	
			Y10		JK Samples		Average			
Part-of-speech Standard:			Nakagawa	Simplified	Nakagawa	Simplified	Nakagawa	Simplified	N-Grams	Term Frequency
Dictionary	JK	P	0.771	0.786	0.965	0.974	0.868	0.880	NO	YES
		R	0.540	0.551	0.965	0.974	0.753	0.763		
		F_1	0.635	0.648	0.965	0.974	0.800	0.811		
		P	0.702	0.718	0.967	0.972	0.835	0.845	YES	NO
		R	0.492	0.503	0.967	0.972	0.730	0.738		
		F_1	0.579	0.592	0.967	0.972	0.773	0.782		
		P	0.794	0.809	**0.977**	**0.981**	0.886	0.895	YES	YES
		R	0.556	0.567	**0.977**	**0.981**	0.767	0.774		
		F_1	0.654	0.667	**0.977**	**0.981**	0.816	0.824		
	KK	P	0.821	0.859	0.713	0.763	0.767	0.811	NO	YES
		R	0.807	0.845	0.563	0.603	0.685	0.724		
		F_1	0.814	0.852	0.629	0.674	0.722	0.763		
		P	0.853	0.886	0.666	0.737	0.760	0.812	YES	NO
		R	0.840	0.872	0.526	0.582	0.683	0.727		
		F_1	0.847	0.879	0.588	0.650	0.717	0.765		
		P	0.859	0.891	0.728	0.790	0.794	0.841	YES	YES
		R	0.845	0.877	0.575	0.624	0.710	0.751		
		F_1	0.852	0.884	0.643	0.697	0.747	0.791		
	JK+KK	P	0.855	0.876	0.960	0.970	0.908	0.923	NO	YES
		R	0.850	0.872	0.960	0.970	0.905	0.921		
		F_1	0.853	0.874	0.960	0.970	0.906	0.922		
		P	0.866	0.892	0.942	0.949	0.904	0.921	YES	NO
		R	0.861	0.888	0.942	0.949	0.902	0.919		
		F_1	0.864	0.890	0.942	0.949	0.903	0.920		
		P	**0.882**	**0.903**	**0.977**	**0.981**	**0.930**	**0.942**	YES	YES
		R	**0.877**	**0.898**	**0.977**	**0.981**	**0.927**	**0.940**		
		F_1	**0.880**	**0.901**	**0.977**	**0.981**	**0.928**	**0.941**		

Table 5.15: A sentence from the JK dictionary processed by POST-AL, with POS annotations (second line) and word-to-word translation into Japanese (fourth line).

POST-AL output:	iyosno ku hosipire kusne na 副詞 人称接辞 他動詞 助動詞 終助詞 [Adverb Personal affix Transitive verb Aux. verb Final particle] 最後に/後で 私は/私が/私の 返す つもりである よ/か ['the end'/'later' 'I'/'my' 'return' 'intend' EMPHASIS]
Meaning:	"I'll return it later"

5.2 MiNgMatch – Fast N-gram Model for Word Segmentation

One way to handle ambiguity – a major challenge in any Natural Language Processing task – is to consider the target text in context. A typical approach is to use an n-gram model, where the probability of a word depends on the $n-1$ previous words. In this section, I argue that in the context of word segmentation, the problem can be reduced to finding the shortest sequence of n-grams matching the input text, with little or no drop in performance compared to state-of-the-art methodologies. In order to verify the usability of my approach, I compare it with three other segmenters, including state-of-the-art lexical n-gram models and a neural model performing word segmentation in the form of character sequence labelling.

Word segmentation is a part of the process of tokenization, a preprocessing stage present in a wide range of higher level Natural Language Processing tasks (such as part-of-speech tagging, entity recognition and machine translation), where the text is divided into basic meaningful units (referred to as *tokens*), such as words and punctuation marks. In the case of writing systems using explicit word delimiters (e.g., whitespaces), tokenization is considered a trivial task. However, sometimes the information about word boundaries is not encoded in the surface form (as in Chinese script), or orthographic words are too coarse-grained and need to be further analyzed – which is the case for many texts written in Ainu. In order to effectively process such texts, one needs to identify the implicit word boundaries.

5.2.1 Related Work

Existing approaches to the problem of tokenization and word segmentation can be largely divided into rule-based and data-driven methods. Data-driven systems may be further subdivided into lexicon-based systems and those employing statistical language models or machine learning.

In space-delimited languages, rule-based tokenizers – such as the Stanford Tokenizer[2] [77] – are sufficient for most applications. On the other hand, in

[2] https://nlp.stanford.edu/software/tokenizer.html

languages where word boundaries are not explicitly marked in text (such as Chinese and Japanese), word segmentation is a challenging task, receiving a great deal of attention from the research community. For such languages, a variety of data-driven word segmentation systems have been proposed. Among dictionary-based algorithms, one of the most popular approaches is the longest match method (also referred to as the maximum matching algorithm or MaxMatch) [153] and its variations [91, 129]. In more recent work, however, statistical and machine learning methods prevail [110, 109, 112, 70, 97]. Furthermore, as in many other Natural Language Processing tasks, the past few years have witnessed an increasing interest towards artificial neural networks among the researchers studying word segmentation, especially for Chinese. A substantial part of the advancements in this area stem from using large external resources, such as raw text corpora, for pretraining neural models [111, 22, 155, 156, 73]. Unfortunately, such large-scale data is not available for many lesser-studied languages, including Ainu. For Japanese and Chinese, word segmentation is sometimes modelled jointly with part-of-speech tagging, as the output of the latter task can provide useful information to the segmenter [70, 160, 89, 138].

Outside of the East Asian context, word segmentation-related research is focused mainly on languages with complex morphology and/or extensive compounding – such as Finnish, Turkish, German, Arabic and Hebrew – where splitting coarse-grained surface forms into smaller units leads to a significant reduction in the vocabulary size and thus lower proportion of out-of-vocabulary words [88, 74, 13, 147, 33]. Apart from that, even in languages normally using explicit word delimiters, there exist special types of text specific to the web domain, such as Uniform Resource Locators (URL) and *hashtags*, whose analysis requires the application of a word segmentation procedure [151, 33].

In 2016 Grant Jenks released WordSegment – a Python module for word segmentation, utilizing a Stupid Backoff model[3]. Due to relatively low computational cost, Stupid Backoff [17] is good for working with extremely large models, such as the Google's trillion-word corpus[4] used as WordSegment's default training data. In terms of the model's accuracy, however, other language modelling methods – in

[3] http://www.grantjenks.com/docs/wordsegment/
[4] https://ai.googleblog.com/2006/08/all-our-n-gram-are-belong-to-you.html

particular the approach proposed by Kneser and Ney [67] and enhanced by Chen and Goodman [23] – proved to perform better, especially with smaller amounts of data [17]. For that reason, in this research, apart from comparing my word segmentation algorithm to WordSegment, I carried out additional experiments with a segmentation algorithm based on an n-gram model with modified Kneser-Ney smoothing. In the context of word segmentation, Kneser-Ney smoothing has previously been used by Doval and Gómez-Rodríguez [33].

Apart from models concerned directly with words, a widely practised approach to word segmentation is to define it as a character sequence labelling task, where each character is assigned with a tag representing its position in relation to word boundaries. While the early works belonging to this category relied on "traditional" classification techniques, such as maximum entropy models [154] and Conditional Random Fields [159], in recent studies neural architectures are being actively explored [160, 111, 138, 73, 52]. In 2018, Shao et al. [137] released a language-independent character sequence tagging model based on recurrent neural networks with Conditional Random Fields interface, designed for performing word segmentation in the Universal Dependencies framework. It obtained state-of-the-art accuracies on a wide range of languages. One of the key components of their methodology (originally proposed in [138]) are the concatenated n-gram character representations, which offer a significant performance boost in comparison to conventional character embeddings, without resorting to external data sources. I used their implementation in the experiments described later in this section, in order to verify how a character-based neural model performs under extremely low-resource conditions, such as those of the Ainu language, and how it compares with segmenters utilizing lexical n-grams, including ours.

To address the problem of word segmentation in the Ainu language, Ptaszynski and Momouchi [119] proposed a segmenter based on the longest match method. Later, Ptaszynski et al. [117] investigated the possibility of improving its performance by expanding the dictionary base used in the process. Nowakowski et al. [102] developed a lexicon-based segmentation algorithm maximizing mean token length. Finally, Nowakowski et al. [104] proposed a segmenter searching for the minimal sequence of n-grams matching the input string, an early and less efficient version of the MiNgMatch algorithm presented in this section.

5.2.2 Description of the Proposed Approach

In the proposed method, I reduce the problem of word segmentation to that of finding the shortest sequence of lexical n-grams matching the input string. For each space-delimited segment in the input text, the algorithm finds a single n-gram or the shortest sequence of n-grams, such that after concatenation and removing all whitespaces is equal to that input segment. In cases where multiple segmentation paths with the same number of n-grams are possible, the sequence with the highest score is selected. Scoring function, given a candidate sequence S, can be defined as below:

$$Score(S) = \prod_{s \in S} \frac{Count(s)}{N} \tag{5.8}$$

where $Count(\cdot)$ denotes the frequency of a particular n-gram in the training corpus and N is the total number of n-grams in that corpus not exceeding the maximum n-gram order specified for the model.

If the model is unable to match any n-gram to the given string or its part, it is treated as an out-of-vocabulary item and returned without modification. Furthermore, the user may specify the maximum number of n-grams to be used in the segmentation of a single input segment. Strings for which the algorithm could not match a sequence of n-grams equal to or shorter than the limit, are retained without modification. The only exception to that rule are punctuation marks – they are separated from alpha-numeric strings in a post-processing step.

5.2.2.1 N-gram Data

Listing 5.5 shows a sample from the data used by the model. The first column contains unsegmented strings used by the matching algorithm, each of them corresponding to a lexical n-gram. In the rightmost column, I store precomputed scores, represented as logarithms.

Once the best sequence of n-grams has been selected, the indices of word boundaries for each n-gram (stored in the second column) are used to produce the final segmentation. For cases where multiple n-gram patterns recorded in the training corpus resulted in the same string after removing whitespaces from between the tokens, I only included the most frequent segmentation in the model. For instance, the 3-gram *aynu mosir ka* ("the land of Ainu also") – which appeared

in the data 6 times – was pruned, as the bigram variant *aynumosir ka* was more frequent, with 63 occurrences. An alternative segmentation can still be returned by the segmenter if it is more frequent in a longer context. For example, although the preferred segmentation of the string *ciki* is *ciki* ("if"; 594 instances), rather than *ci ki* (first person pronominal marker *ci* attached to auxiliary verb *ki*) with 32 occurrences, in the case of a longer segment: *cikisiri*, the only segmentation attested in the data is *ci ki siri* (*ci ki* followed by the nominalizing evidential particle *siri*), appearing 3 times in the training corpus.

Listing 5.5: Sample from the n-gram data used by MiNgMatch Segmenter.

N-gram (unsegmented):	*Indices of word boundaries:*	*Score:*
koranan	5,3	-3.34220648264453
penekusu	4,2	-3.3427874786936
korwa	3	-3.3439518077607
patek		-3.34628987252967

5.2.2.2 Computational Cost

The maximum number of candidate segmentations to be generated by the algorithm, given a string composed of n characters, can be calculated as follows:

$$\sum_{k=0}^{min\{m,l\}} \binom{n-1}{k} \tag{5.9}$$

where m stands for the smallest number of n-grams existing in the model needed to create a sequence matching the input string, and l represents the limit of n-grams per input string (specified by the user), such that $l \leq n$. In practice it means that, apart from rare situations where only a sequence of single-character unigrams can be matched to the given string, my algorithm has a lower computational cost than a model which considers all the 2^{n-1} possible segmentations[5].

[5]Obviously, a word segmentation algorithm evaluating each unique segmentation path would be highly impractical; a typical approach, also taken by myself, is to reduce that number by applying dynamic programming and memoization techniques.

5.2.3 Data and Experiments

5.2.3.1 Training Data

Language models applied in this research were trained on Ainu language textual data from eight different sources (seven of them were previously included in my corpus, and therefore are described in details in Chapter 4):

(A) *Ainu Shin'yōshu* [29] (**SYOStrain**)

I only included 11 epics in the training set, while the remaining 2 texts were used as test data (see the next section).

(B) A Talking Dictionary of Ainu: A New Version of Kanazawa's Ainu Conversational dictionary [21] (**TDOA**)

For training, I used the modernized transcription produced by Bugaeva et al. [21]. The last 285 entries (roughly 10% of the dictionary, character-wise) were excluded from the training data, in order to use them as test data in evaluation experiments (see the next section).

(C) Glossed Audio Corpus of Ainu Folklore [95] (**GACF**)

(D) Dictionary of Ainu place names [87] (**MOPL**)

(E) Dictionary of the Mukawa dialect of Ainu [24] (**MUKA**)

(F) Collection of Ainu Oral Literature [98] (**NIBU**)

(G) Ainu Language Archive – Materials [145] (**AASI**)

(H) Ainu Language Archive – Dictionary [145] (**AAJIhw**)

An online lexicon consisting of digitized versions of three Ainu-Japanese dictionaries [26, 60, 143], comprising a total of 33,126 entries. I used only the headwords included in the dictionary. After removing duplicates (homographic entries), a total of 16,107 entries remained.

The following post-processing steps were applied to the training corpus:

(1) The data was cleaned, resulting in files containing only raw Ainu text in Latin alphabet.

(2) Accented vowels (⟨á⟩, ⟨é⟩, ⟨í⟩, ⟨ó⟩, ⟨ú⟩) used in some materials were replaced with their unaccented counterparts (⟨a⟩, ⟨e⟩, ⟨i⟩, ⟨o⟩, ⟨u⟩).

(3) Underscores (⟨_⟩) used in some materials to indicate phonological alternations were removed.

(4) Equality signs (⟨=⟩) used to denote personal markers were either removed, or replaced with whitespaces (if there was no whitespace in the original text). While their presence is an unambiguous indicator of a boundary between two tokens[6], they were not used in older texts, which are going be the main target of a word segmentation system, therefore I decided to exclude them from the data. This resulted in a corpus of text comprising a total of 481,291 segments (space-delimited units of text). The statistics of all eight datasets after this step are shown in Table 5.17.

(5) Finally, punctuation marks were separated from words. However, non-alphanumeric characters used word-internally (e.g., hyphens indicating boundaries between the constituents of compound words and apostrophes representing glottal stop) were not modified.

Table 5.17: Statistics of Ainu text collections and dictionaries used as the training data.

Text Collection:	SYOStrain	TDOA	GACF	MOPL	MUKA	NIBU	AASI	AAJIhw	Overall
Characters (excluding spaces):	36,780	54,545	82,436	26,872	317,099	459,253	803,945	131,449	1,912,379
Segments:	8,786	12,978	22,559	9,246	71,232	122,314	218,069	16,107	481,291
Avg. segment length:	4.186	4.203	3.654	2.906	4.452	3.755	3.687	8.161	3.973

5.2.3.2 Test Data

Two different sets of held-out data were used in the evaluation experiments:

[6] Although they are traditionally referred to as "affixes", I treat those morphemes as separate units, which is a common practice among present-day experts. For a detailed analysis of their morphological status, please refer to Bugaeva [20].

(A) *Ainu Shin'yōshu* [29] (**SYOS**)

This dataset consists of the two *yukar* epics from *Ainu Shin'yōshu* which were excluded from the training data:

- "Esaman yaieyukar, 'Kappa reureu kappa'" ("'Kappa reureu kappa', the song sung by the otter");
- "Pipa yaieyukar, 'Tonupeka ranran'" ("'Tonupeka ranran', the song sung by the marsh shellfish").

(B) *Ainugo Kaiwa Jiten* [55] (**AKJ**)

The portion of the original dictionary corresponding to the entries from A Talking Dictionary of Ainu which I removed from the training data, was applied as the second evaluation dataset.

In the test data I retained the word segmentation of the original transcriptions (by Chiri [29] and Jinbō and Kanazawa [55]). However, in order to prevent differences in spelling from affecting the word segmentation algorithm's performance, the text was preprocessed by unifying its spelling with modern versions transcribed by Kirikae [65] and Bugaeva et al. [21]. In the case of the *Ainugo kaiwa jiten* and TDOA, there were also some differences in the usage of punctuation marks as well as several words which appeared in the original text, but the authors of the modernized transcription decided to remove them – in such cases the text was unified with the modern transcription, with the exception of equality signs attached to personal markers, which were omitted. A sample sentence from the *Ainugo kaiwa jiten* before and after this preprocessing step is shown in Table 5.18. Table 5.19 presents the statistics of both evaluation datasets, in comparison with the portions of modernized texts corresponding to them.

5.2.3.3 Experiment Setup

In the experiments I tested the following word segmentation systems:

(1) a corpus-based word segmentation algorithm minimizing the number of n-grams needed to match the input string (MiNgMatch Segmenter[7]);

[7]The implementation is publicly available: https://github.com/karol-nowakowski/MiNgMatchSegmenter.

Table 5.18: A fragment from the Talking Dictionary of Ainu (TDOA)/Ainugo kaiwa jiten (AKJ) dataset.

***Ainugo kaiwa jiten* [55]:**	Nepka ayep an shiri he an?
TDOA [21]:	nep ka a= ye p an siri an?
AKJ:	nepka ayep an siri an?
Translation [21]:	Is there something you (wanted) to say?

Table 5.19: Statistics of the samples used for evaluation and their modern transcription equivalents.

Data	Characters (Excluding Spaces)	Segments	Avg. Segment Length	OoV* Rate
SYOS:	2,667	441	6.070	0.259
Kirikae [65]:		643	4.163	0.034
AKJ:	4,840	1,143	4.234	0.044
Bugaeva et al. [21]:		1,259	3.844	0.005

*** OoV** – out-of-vocabulary words.

(2) a segmentation algorithm with Stupid Backoff language model (WordSegment with modifications);

(3) a segmentation algorithm with a language model applying modified Kneser-Ney smoothing (later referred to as "mKN");

(4) a segmentation system based on character sequence labelling using a neural model [137] (later I will refer to it as "Universal Segmenter").

5.2.3.4 MiNgMatch Segmenter

The algorithm was tested in two variants:

- with the limit of n-grams per input segment equal to the number of characters in the input string;
- with the limit of n-grams per input segment set to 2 (based on the observation that in most cases where a single input segment is divided into 3 or more

n-grams, that segmentation is incorrect).

In experiments conducted by Nowakowski et al. [104] with an early version of the segmenter, the best results were in most cases yielded with the order of n-grams not exceeding 5-grams. Thus, for the n-gram models examined in this section, I set the limit of n to 5.

5.2.3.5 WordSegment (Stupid Backoff Model)

WordSegment is an open source Python module for word segmentation developed by Grant Jenks, based on the work of Peter Norvig [100]. In the evaluation experiments I applied the system with two modifications:

(A) I added the option of using n-gram models with the order of n-grams higher than 2 (in original WordSegment, only unigrams and bigrams were used, whereas I wanted to test models with the order of up to 5).

(B) I added the possibility of manipulating the backoff factor. Although it was a part of the original formulation by Brants et al. [17], Peter Norvig and Grant Jenks omitted it from their implementations.

I examined three different values of the backoff factor:

- 1 (i.e. no backoff factor, as in original WordSegment);
- 0.4 (as suggested by Brants et al. [17]; later I will refer to this model as "SB-0.4");
- 0.09, only applied to 1-grams (this configuration achieved the best F-score in preliminary experiments, at the cost of lower Precision). Let w_i denote a candidate word to be evaluated in the context of k previous words (w_{i-k}^{i-1}). The recursive scoring function employed in this variant (later referred to as "SB-0.09") can be defined as follows:

$$Score(w_i, k) = \begin{cases} \frac{Count(w_{i-k}^{i})}{Count(w_{i-k}^{i-1})} & \text{if } Count(w_{i-k}^{i}) > 0 \\ \begin{cases} Score(w_i, k-1) & \text{if } k \geq 2 \\ \alpha \frac{Count(w_i)}{N_1} & \text{otherwise.} \end{cases} & \text{otherwise.} \end{cases} \tag{5.10}$$

with α representing the backoff factor specified for unigrams, and N_1 being the total number of unigrams in the training corpus.

5.2.3.6 Segmenter with Language Model Applying Modified Kneser-Ney Smoothing

In the next experiment, I tested a word segmentation system similar to WordSegment (the same dynamic programming algorithm is used to generate candidate segmentations), but employing a language model with modified Kneser-Ney smoothing for choosing the most probable segmentation path. The model was generated using the KenLM Language Model Toolkit[8].

Analogically to the experiments with my system and WordSegment, I used language models with the maximum order of n-grams set to 5.

5.2.3.7 Universal Segmenter

Apart from segmenters utilizing language models based on lexical n-grams, I carried out a series of experiments using the character-level sequence labelling model developed by Yan Shao et al. [137]. I trained the model in three different variants:

Default Model (henceforth, "US-Default") In this experiment, I applied the default training settings, designed to work with space-delimited languages. The same training data as in previous experiments was used, which means that tokens in the training set correspond to those in gold standard data.

With Spaces Ignored (henceforth, "US-ISP") Here, I trained the model with the `-isp` argument. It results in the removal of space delimiters from the training data, which means it is effectively treated in a similar way to Chinese or Japanese script.

With Multi-Word Tokens (later referred to as "US-MWTs_rnd" and "US-MWTs") When processing older Ainu texts, many space-delimited segments need to be split in multiple tokens. Consequently, the default model relying on

[8] https://kheafield.com/code/kenlm/

whitespaces and trained on the data with modern segmentation is ineffective. Unlike in Chinese or Japanese, however, a large portion of word boundaries is already correctly indicated by whitespaces in the input text, so ignoring them altogether (as in US-ISP models) is not the optimal method, as well. In order to create a model better suited to the task, I used the concept of multi-word tokens[9] existing in Universal Dependencies and also reflected in the Universal Segmenter.

Firstly, I converted the two datasets (SYOStrain and TDOA) for which both old and modernized transcriptions exist, to a format where boundaries between words grouped together as a single space-delimited string in the original transcription are treated as boundaries between components of a multi-word token. For the remaining six datasets, however, only a single transcription by contemporary experts is available. I therefore applied the following two methods to simulate sparser word segmentation of old texts by generating multi-word tokens artificially:

(A) As a baseline method, I created multi-word tokens in a random manner. Namely, I assigned each whitespace in the data with a 50% chance of being removed and thus becoming a boundary between components of a multi-word token. This resulted in the generation of 105,663 multi-word tokens. Later I will refer to the models learned from this version of the training data as "US-MWTs_rnd".

(B) In the second approach, multi-word tokens were generated in a semi-supervised manner using the Universal Segmenter itself. To achieve that, I converted multi-word tokens previously identified in SYOStrain and TDOA to multi-token words (defined in the UD scheme as words consisting of multiple tokens, but treated as a single syntactic unit) and trained a word segmentation model on these two datasets. The resulting model was then used to process the rest of the training corpus. As a result, some tokens were grouped in multi-token words (a total of 70,373 such words were generated). In the final step, I converted the multi-token words to multi-word tokens. This variant of the data was used to train the group of models later referred to as "US-MWTs".

I illustrate the operations described above in Table 5.20, using a sample from the SYOStrain dataset.

[9] https://universaldependencies.org/u/overview/tokenization.html

Table 5.20: Operations on training data for the Universal Segmenter.

Original text [29]:

Shineanto ta shirpirka kusu [...]
("One day, since the weather was nice [...]")

Modernized transcription [65]:

Sine an to ta sir pirka kusu [...]

Training data (in CoNLL-U format*) with multi-word tokens:

```
1-3 sineanto _ _ _ _ _ _ _ _ _
   1 sine _ _ _ _ _ _ _ _ _
   2 an _ _ _ _ _ _ _ _ _
   3 to _ _ _ _ _ _ _ _ _
   4 ta _ _ _ _ _ _ _ _ _
5-6 sirpirka _ _ _ _ _ _ _ _ _
   5 sir _ _ _ _ _ _ _ _ _
   6 pirka _ _ _ _ _ _ _ _ _
   7 kusu _ _ _ _ _ _ _ _ _
```

Training data (in CoNLL-U format*) with multi-token words:

```
1 sine an to _ _ _ _ _ _ _ _
   2 ta _ _ _ _ _ _ _ _ _
3 sir pirka _ _ _ _ _ _ _ _
   4 kusu _ _ _ _ _ _ _ _
```

* See: https://universaldependencies.org/format.html.

Apart from simple character embeddings, the Universal Segmenter allows the usage of concatenated n-gram vectors encoding rich local information. I investigated the performance with 3-, 5-, 7-, 9- and 11-grams. Any parameters of the training process not mentioned above were set to default values.

5.2.3.8 Evaluation Method

In order to evaluate word segmentation performance, I employed three metrics: Precision (P), Recall (R) and balanced F-score (F_1). Precision is defined as the proportion of correct word boundaries (whitespaces) within all word boundaries returned by the system (B_s), whereas Recall is the portion of word boundaries present in expert-annotated data (B_e) which were also correctly predicted by the segmenter. The balanced F-score is the harmonic mean of Precision and Recall.

$$P = \frac{|B_s \cap B_e|}{|B_s|} \tag{5.11}$$

$$R = \frac{|B_s \cap B_e|}{|B_e|} \tag{5.12}$$

$$F_1 = 2\frac{PR}{P+R} \tag{5.13}$$

In addition, I evaluated word-level Accuracy for OoV words, defined as the proportion of unseen tokens in expert-annotated data (U_e), correctly segmented by the system (U_s):

$$Accuracy = \frac{|U_s|}{|U_e|} \tag{5.14}$$

5.2.4 Results and Discussion

The results of the evaluation experiments with the proposed algorithm are presented in Table 5.21. The variant without the limit of n-grams per input segment produces unbalanced results (especially on SYOS), with relatively low Precision. After setting the limit to 2, Precision improves at the cost of a drop in Recall. The F-score is better for SYOS, while on AKJ there is a very slight drop.

Table 5.22 shows the results of experiments with the Stupid Backoff model.

When no backoff factor is applied, results for both test sets are similar to those from the MiNgMatch Segmenter without the limit of n-grams per input segment. Setting the backoff factor to an appropriate value allows for significant improvement in Precision and F-score (and in some cases also small improvements in Recall). For the F-score, it is better to set a low backoff factor (e.g., 0.09) for 1-grams only, than to set it to a fixed value for all backoff steps (e.g., 0.4, as Brants et al. [17] did). A backoff factor of 0.4 gives significant improvement in Precision with higher order n-gram models, but at the same time Recall drops drastically and overall performance deteriorates. For models with an n-gram order of 3 or higher, the backoff factor has a bigger impact on the results than further increasing the order of n-grams included in the model. A comparison with the results yielded by MiNgMatch shows that setting the limit of n-grams per input segment is more effective than Stupid Backoff as a method for improving precision of the segmentation process – it leads to a much smaller drop in Recall.

The results of the experiment with models employing modified Kneser-Ney smoothing are shown in Table 5.23. They achieve higher Precision than both the other types of n-gram models. Nevertheless, due to very low Recall, the overall results are low.

The results obtained by the Universal Segmenter are presented in Table 5.24. The default model (regardless of what kind of character representations are used – conventional character embeddings or concatenated n-gram vectors) learns from the training data that the first and the last character of a word (corresponding to `B`, `E` and `S` tags) are always adjacent either to the boundary of a space-delimited segment or to a punctuation mark. As a result, the model separates punctuation from alpha-numeric strings found in the input, but never applies further segmentation to them.

US-ISP models are better but still notably worse than lexical n-gram models (especially on SYOS). Unlike with default settings, the model trained on data without whitespaces learns to predict word boundaries within strings of alpha-numeric characters. However, when presented with test data including spaces, they impede the segmentation process rather than supporting it. As shown in Table 5.25, if I only take into account the word boundaries not already indicated in the raw test set, the model makes more correct predictions in data where the whitespaces

Table 5.21: Evaluation results – MiNgMatch Segmenter (best results in bold).

Max. N-gram Order:			**2**		**3**		**4**		**5**	
	Test Data:		**SYOS**	**AKJ**	**SYOS**	**AKJ**	**SYOS**	**AKJ**	**SYOS**	**AKJ**
Max. n-grams per segment	**Max.***	P	.918	.968	.923	.969	.925	.969	.923	.969
		R	.918	.986	.943	.989	.952	.989	.952	.990
		F_1	.918	.977	.933	.979	.938	.979	.938	**.980**
	2	P	.952	.972	.955	.972	.957	.972	.956	.973
		R	.899	.981	.924	.985	.933	.985	.933	.986
		F_1	.925	.976	.939	.978	**.945**	.979	.944	.979

*** Max.** – number of characters in the input segment.

Table 5.22: Evaluation results – Stupid Backoff model (best results in bold).

Max. N-gram Order:			**2**		**3**		**4**		**5**	
	Test Data:		**SYOS**	**AKJ**	**SYOS**	**AKJ**	**SYOS**	**AKJ**	**SYOS**	**AKJ**
Backoff factor	**1**	P	.915	.965	.923	.965	.923	.965	.923	.965
		R	.916	.986	.958	.989	.961	.988	.961	.988
		F_1	.915	.975	.940	**.977**	.942	.976	.942	.976
	0.4	P	.924	.967	.934	.968	.952	.972	.958	.974
		R	.921	.988	.944	.986	.913	.979	.875	.966
		F_1	.922	**.977**	.939	**.977**	.932	.975	.915	.970
	n=1: **0.09** n>1: **1**	P	.934	.968	.937	.968	.937	.965	.937	.965
		R	.927	.986	.961	.987	.963	.986	.963	.986
		F_1	.930	**.977**	.949	**.977**	**.950**	.975	**.950**	.975

have all been removed.

Table 5.25: US-ISP model (with 9-gram vectors): F-score for word boundaries not indicated in original transcription.

Spaces in Test Data Retained	**Test Data**	
	SYOS	**AKJ**
YES	.811	.880
NO	.844	.930

Models with multi-word tokens achieve significantly higher results. Precision of the US-MWTs model is on par with the segmenter applying Kneser-Ney smoothing,

Table 5.23: Evaluation results – model with Kneser-Ney smoothing (best results in bold).

Max. N-gram Order:	**2**		**3**		**4**		**5**	
Test Data:	**SYOS**	**AKJ**	**SYOS**	**AKJ**	**SYOS**	**AKJ**	**SYOS**	**AKJ**
P	.970	.979	.975	.978	.975	.979	.975	.979
R	.693	.946	.724	.944	.726	.943	.726	.943
F_1	.808	**.962**	.831	.961	**.832**	.961	**.832**	.961

Table 5.24: Evaluation results – Universal Segmenter (best results in bold).

Model Version:			**Default**		**ISP**		**MWTs_rnd**		**MWTs**	
Test Data:			**SYOS**	**AKJ**	**SYOS**	**AKJ**	**SYOS**	**AKJ**	**SYOS**	**AKJ**
Order of concatenated n-gram vectors	–	F_1	.809	.931	.874	.950	-	-	-	-
	3	F_1	.813	.931	.894	.946	.939	.971	.938	.974
	5	F_1	.812	.931	.888	.958	.945	.973	.948	.978
	7	F_1	.812	.931	.897	.955	.944	.973	.948	.977
	9	P	.998	.981	.910	.952	.950	.972	.976	.980
		R	.686	.885	.910	.962	.947	.976	.927	.979
		F_1	.813	.931	.910	.957	.948	.974	**.951**	**.979**
	11	F_1	.812	.931	.914	.954	.946	.974	.950	.975

while maintaining relatively high Recall. It yields lower Recall than the model with randomly generated multi-word tokens, but the F-score is higher due to better Precision.

With the exception of the US-ISP model on SYOS, all variants of the neural segmenter achieved the best performance with concatenated 9-gram vectors. This contrasts with the results reported by Shao et al. [138] for Chinese, where in most cases there was no further improvement beyond 3-grams. This behavior is a consequence of differences between writing systems: words in Chinese are on average composed of less characters than in languages using alphabetic scripts. Due to a much bigger character set size, *hanzi* characters are also more informative to word segmentation [137], hence better performance with models using shorter context.

5.2.4.1 General Observations

Due to data sparsity, n-gram coverage in the test set (the fraction of n-grams in the test data that can be found in the training set) is low (see Table 5.26). It means that many multi-word tokens from the test set are known to n-gram models as separate unigrams, but not in the form of a single n-gram. The Stupid Backoff model with a backoff factor for unigrams set to a moderate value (such as 0.09) is able to segment such strings correctly. However, it also erroneously segments some OoV single-word tokens whose surface forms happen to be interpretable as a sequence of concatenated in-vocabulary unigrams, resulting in lower Precision. On the other hand, models assigning low scores to unigrams (such as a 4- or 5-gram model with the Stupid Backoff and backoff factor set as suggested by Brants et al. [17], and in particular the model applying modified Kneser-Ney smoothing) are better at handling OoV words (see Table 5.27), but as a result of probability multiplication, in many cases they score unseen multi-word segments higher than a sequence of unigrams into which the given segment should be divided, hence yielding lower Recall.

Table 5.26: N-gram coverage.

N-gram Order:	**1**	**2**	**3**	**4**	**5**
Test data:	**N-gram coverage:**				
SYOS	.966	.627	.338	.188	.128
AKJ	.995	.792	.487	.236	.099

Table 5.27: Word-level Accuracy for OoV words (best models only).

Model:	**MiNgMatch**	**SB-0.4**	**SB-0.09**	**mKN**	**US-MWTs**
Test data:	**Accuracy:**				
SYOS	.320	.120	.000	.320	.560
AKJ	.000	.000	.000	.125	.625

Universal Segmenter operates at the level of characters rather than words, which makes it more robust against unseen words. This, along with the ability of

neural models to transform discrete, sparse inputs into continuous representations capturing similarities between them, such as morphological features [33], explains the fact that it is able to achieve high Precision while maintaining relatively high Recall.

In line with these observations, I found Universal Segmenter to be the only segmenter in the experiments whose output includes tokens seen neither in the training data nor in the test set. For instance, it correctly segmented the input token *ekampaktehi* into *ekampakte hi* ("a promise"), whereas other systems either did not divide it at all, or segmented it into a sequence of in-vocabulary unigrams (e.g., *ekampak te hi*).

5.2.4.2 Error Comparison

Using the outputs of the best performing models, I measured how similar the errors made by different segmenters were. In particular, I calculated the Jaccard index between lists of errors found in each pair of outputs.

Results are presented in Table 5.28. Output of the model with modified Kneser-Ney smoothing is the least similar to most other models' outputs, which can be explained simply by the fact that it made the highest number of errors on both datasets (statistics are shown in Table 5.29). On the other hand, the Universal Segmenter's output, while containing numbers of errors comparable to those produced by the best performing n-gram models, also exhibits a low level of similarity to them.

Table 5.28: Error comparison (Jaccard index).

Test set: SYOS	**MiNgMatch**	**SB-0.09**	**mKN**
US-MWTs	.220	.153	.195
mKN	.176	.159	
SB-0.09	.505		
Test set: AKJ	**MiNgMatch**	**SB-0.09**	**mKN**
US-MWTs	.414	.390	.474
mKN	.369	.355	
SB-0.09	.714		

Table 5.29: Statistics of word segmentation errors.

Model:	**MiNgMatch**	**SB-0.09**	**mKN**	**US-MWTs**
Test data:		**Errors:**		
SYOS	71	66	189	62
AKJ	50	58	91	49

Indeed, qualitative analysis of segmentations generated by the neural model confirms that in some parts they are quite different from the predictions made by other models. For instance, the two segments *wenpuri enantuykasi* were correctly divided into *wen puri enan tuykasi* ("[her] face [took] the color of anger") only by the Universal Segmenter. All other models incorrectly split the word *tuykasi* (possessive form of the locative noun *tuyka*, meaning "on [the face]") into *tuyka si*, the reason being the fact that the n-gram *wen puri enan tuyka* is attested (with 4 instances) in the training set. Conversely, there are also some errors only made by the Universal Segmenter. For instance, it was the only system to divide the in-vocabulary word *ayapo* (exclamation of pain) into two tokens: *a* and *yapo*, out of which *yapo* does not appear in the training data. Another example is the phrase *ki aineno* ("eventually"), transcribed by Kirikae as *ki a ine no* (3 instances in the training set) and segmented in the same way by n-gram models, whereas the neural model treated the last two words as a single unit, *ineno*. This prediction, however, might be arguably considered correct, as there exists one instance of *ineno* in Kirikae's transcription, used in the same context (*iki a ineno*). Based on the observations described above, I believe that implementing an ensemble of an n-gram model and a character sequence labelling neural model shall be an interesting avenue for future work.

5.2.4.3 Results on SYOS with Two Gold Standard Transcriptions

As mentioned in Section 2.2.2, there is a certain amount of inconsistencies in word segmentation even between contemporary scholars of Ainu, which means they are also present in the data. With that in mind, I decided to cross-check the results of the experiments against an additional gold standard transcription. For that purpose I used an alternative modernized transcription of SYOS by Katayama [56].

Firstly, I compared Katayama's transcription with the version edited by Kirikae [65], using the same evaluation metrics as in previous experiments with segmentation algorithms. The results are presented in Table 5.30.

Table 5.30: Katayama's transcription evaluated against Kirikae's transcription.

Precision	**Recall**	**F-score**
.979	.941	.960

My assumption was that – in spite of making different decisions as to whether to group certain morphemes together or to treat them as separate units – both experts produced correct transcriptions. In order to investigate the effect of this phenomenon on the experiments, I re-evaluated the outputs of the best performing segmentation models using a combination of both experts' transcriptions as the gold standard data. This time, Precision was defined as the proportion of word boundaries predicted by the model that can be also found in either of the gold standard transcriptions:

$$P = \frac{|B_s \cap (B_{e_1} \cup B_{e_2})|}{|B_s|} \tag{5.15}$$

Analogically, Recall was defined as the proportion of word boundaries found in both variants of the gold standard which were also correctly predicted by the model:

$$R = \frac{|B_s \cap B_{e_1} \cap B_{e_2}|}{|B_{e_1} \cap B_{e_2}|} \tag{5.16}$$

Results are shown in Table 5.31. Apart from the model with Kneser-Ney smoothing, the results achieved by all models improved substantially. The highest gain was obtained for the proposed algorithm – the result improved to such an extent that it ranked first in terms of F-score. A large share of that difference can be attributed to a single token, *awa* (a conjunction created by combining the perfective aspect marker *a* and a coordinative conjunctive particle *wa*), appearing a total of 17 times in the test set. The MiNgMatch algorithm, operating at the level of input segments, followed Katayama in not splitting *awa*, as it is more frequent in the training data, with 289 occurrences, than the 2-gram variant *a wa* (181

instances). Nevertheless, models considering a wider context preferred the latter option, which conforms with how Kirikae transcribed it.

Table 5.31: SYOS: outputs of the best models re-evaluated against combined gold standard data (Kirikae [65] + Katayama [56]). Best results in bold.

Model:	**MiNgMatch**	**SB-0.4**	**SB-0.09**	**mKN**	**US-MWTs**
Precision	.965	.969	.949	**.981**	.980
Recall	.954	.880	**.964**	.718	.933
F-score	**.959**	.922	.956	.829	.956

5.2.4.4 Execution Speed

Table 5.32 compares the total time taken by each of the best performing models to process the two test sets. In the case of segmenters based on lexical n-grams, I used 5-gram models. The Universal Segmenter's speed was evaluated on the model trained with concatenated 9-gram vector representations. Experiments with n-gram models were carried out on a Windows machine with Intel Core i7 running at 1.90 GHz and 16 GB of RAM. The Universal Segmenter was tested on an Ubuntu machine with four GPUs (NVIDIA GeForce GTX 1080 Ti) and 128 GB of RAM. Each value represents an average of five consecutive runs. The results indicate that the proposed algorithm is unrivalled in terms of speed.

Table 5.32: Execution times in seconds.

Model:	**MiNgMatch**	**SB-0.09**	**mKN**	**US-MWTs**
Time (s)	0.812	5.837	4.785	7.717

5.2.5 Conclusions

In this section, I introduced the MiNgMatch Segmenter: a data-driven word segmentation algorithm finding the minimal sequence of n-grams needed to match the input text. I compared my algorithm with segmenters utilizing two state-of-the-art n-gram language modelling techniques (namely, the Stupid Backoff model and a

model with modified Kneser-Ney smoothing), as well as a neural model performing word segmentation as character sequence labelling.

The evaluation experiments revealed that the proposed approach is capable of achieving overall results comparable with the other best-performing models, especially when one takes into account the variance in notation of certain lexical items by different contemporary experts. Given its low computational cost and competitive results, I believe that MiNgMatch could be applied to other languages, and possibly to other Natural Language Processing problems, such as speech recognition.

In terms of precision of the segmentation process and accuracy for out-of-vocabulary words, the sequence labelling neural model turned out to be the best option. In order to achieve that, however, it needs to be presented with training data tailored to the task, closely mimicking the intended target data. To this end, I demonstrated that such data can be bootstrapped from a small amount of manually annotated text, using the Universal Segmenter itself.

5.3 Applying Support Vector Machines and Neural Models to Part-of-speech Tagging

In this section, I describe an experiment comparing the performance of three different automatic part-of-speech taggers on Ainu language data.

5.3.1 Related Work

The first part-of-speech tagging tool developed for the Ainu language was POST-AL (see Section 5.1). Unlike POST-AL, which performs part-of-speech disambiguation based on a lexicon, state-of-the-art part-of-speech taggers developed for other languages typically utilize part-of-speech annotated language corpora as their training data. One of such tools is the SVMTool, an open source generator of sequential taggers based on Support Vector Machines, developed by Giménez and Márquez [44]. It achieves an accuracy of 97.2% in part-of-speech tagging of English, but has also been applied in studies dedicated to low-resource languages, such as the ones by Hagemeijer et al. [47] and Behera et al. [8]. In recent years, however, Artificial Neural Network-based approaches prevail [e.g., 152, 116, 38]. In this case, word representations pre-trained on a large corpus of raw text (using tools such as word2vec [82, 83, 84] or fastText [16]) have shown to improve the performance. In the context of low-resource languages, where large labelled corpora are difficult to obtain, many studies try to leverage annotations projected from resource-rich languages through word alignments inferred from parallel corpora [e.g., 157, 142, 37, 3, 115], or through cross-lingual word representations [e.g., 158, 38].

5.3.2 Tagging Models Used

In the experiments, I compared POST-AL with two other taggers: the SVMTool and a neural tagger. Specifically, I used the variant of POST-AL with hybrid approach to part-of-speech disambiguation, which yielded the best results in the experiments described in Section 5.1, and the SVMTool v. 1.3.2 (Perl version). The third tagger was a neural network with a single hidden layer.

5.3.2.1 SVMTool Settings

Model settings Training parameters of the SVMTool were set to default values. Appendix A explains the feature set used in each variant of the model applied in this research.

Preliminary experiments revealed that the model assigns tags corresponding to punctuation marks (e.g. ":") to many lexical OoV words. To avoid such behavior, I modified one of the model files containing the list of tags to be considered for OoV tokens, removing such tags from the list.

Tagging parameters In the experiments with the SVMTool-generated tagger, I investigated the performance with different values of the following parameters :

- Tagging strategy (`-T`) – different strategies apply different tagging schemes (greedy or sentence-level) and different variants of the tagging model are used;
- Tagging direction (`-S`) – `LR` (left-to-right), `RL` (right-to-left) or `LRL` (both directions combined). According to Giménez and Márquez [45], tagging direction "varies results yielding a significant improvement when both are combined".

5.3.2.2 Neural Tagger and Word Embeddings

Hyperparameters of the neural network used in the experiment are presented in Table 5.33. The model takes as input the target word along with two words to its left and two words to its right side, and produces the probability distribution over all possible part-of-speech tags. The tagger's embedding layer was initialized with four different types of word representations:

1. Baseline: **one-hot encodings**, where each word is represented as a sparse vector with only one non-zero value, in a vector space of dimension equal to the size of the vocabulary used.
2. **word2vec**[10] [82, 83, 84] – in this approach, a neural network is trained to predict words based on their context (which is known as the Continuous Bag

[10] https://code.google.com/p/word2vec/

of Words model) or vice versa (the Skip-gram model). The feature vectors learned by the network are then used as word representations that capture syntactic and semantic relationships. In this experiment, I employed the Skip-gram model to pre-train 300-dimensional word embeddings on the Ainu corpus described in Chapter 4.

3. **fastText**[11] [16] – an extension of word2vec, which exploits subword information by representing words in terms of character n-grams they consist of. This is especially useful in morphologically rich languages, such as Ainu, where it allows for generating meaningful vector representations also for rare forms and words not seen in the training corpus. In this experiment, I applied 300-dimensional fastText embeddings pre-trained on the Ainu corpus described in Chapter 4.

4. **Cross-lingual embeddings**, derived from the Ainu-Japanese parallel corpus, according to the following procedure: firstly, the Japanese side of the corpus was analyzed with a morpho-syntactic tagger (MeCab[12]) and a part-of-speech tag was attached to each token. Secondly, word alignments were trained, using the GIZA++ toolkit[13] [108]. As a result, for each Ainu word I obtained a list of possible Japanese equivalents (each of them consisting of a word and its part-of-speech tag), along with translation probabilities. An excerpt from the GIZA++'s output is presented in Listing 5.6. Finally, each Ainu word was represented as a vector in a vector space of dimension equal to the size of the Japanese part-of-speech tagset used in MeCab (68 tags), where each vector component represents the probability of the given word for being translated to Japanese using a word assigned with the corresponding tag. Let $p(j, t|a)$ denote the probability that an Ainu word a translates to a word j in the Japanese vocabulary J, assigned with the tag t from the Japanese part-of-speech tagset T. Each word in the Ainu vocabulary is then associated with a vector $\mathbf{v} \in \mathbb{R}^{|T|} = (v_t)_{t \in T}$, where:

[11] https://fasttext.cc/
[12] https://taku910.github.io/mecab/
[13] https://github.com/moses-smt/giza-pp

$$v_t = p(t|a) = \sum_{j \in J} p(j, t|a) \tag{5.17}$$

In contrast to previous studies in cross-lingual neural part-of-speech tagging [e.g., 38, 115], where morpho-syntactic information transferred from a source language was used to generate new training samples for a target language tagger, in the proposed approach it is encoded in the vector representations of Ainu words. The advantage of this method is that it can be combined with existing target language annotations utilizing a language-specific, fine-grained tagset, without the need for mapping them to a simplified, shared tagset[14].

Listing 5.6: Excerpt from the lexical translation probability table generated by GIZA++. Japanese words and their morpho-syntactic tags are separated with "#".

```
kokewrototke　轟き#動詞-自立　0.00109793
kokewrototke　鳴り#動詞-自立　0.0115893
usay　いったい#副詞-一般　1
ka　いったいぜんたい#副詞-一般　9.50356e-06
imu　イム#名詞-固有名詞-人名-姓　0.115427
ki　放り#動詞-自立　2.09294e-09
ram'ositciwre　落ち着い#動詞-自立　0.393449
koyaytaraye　佩い#動詞-自立　1
emus　刃#名詞-一般　0.00444338
emus　佩い#動詞-自立　5.08458e-07
taanpe　こいつ#名詞-一般　0.0268675
koeyapkir　放り#動詞-自立　0.78706
```

Table 5.34 presents the parameters of all four types of word representations used in the experiment.

Code of the neural tagger (the variant using cross-lingual word representations) is included in Appendix B.

[14]In recent years, the majority of studies in cross-lingual part-of-speech tagging utilize the Universal Part-of-Speech Tagset [114] consisting of 12 categories. In Ainu studies, on the other hand, most experts employ more sophisticated part-of-speech classification systems, reflecting language-specific subcategories such as postpositive adverbs and locative nouns. A system using simplified tagset might be insufficient for advanced linguistic analyses.

Table 5.33: Hyperparameters of the neural part-of-speech tagger.

Hyperparameter	Value
Hidden layers	1
Hidden units	50
Learning rate	0.001
Batch size	128
Epochs	10

Table 5.34: Parameters of word representations used with the neural tagger.

	One-hot	word2vec	fastText	Cross-ling.
Dimension	vocab. size	300	300	68
Pre-training task	N/A	Skip-gram	Skip-gram	N/A
Window size	N/A	3	3	N/A
Min. count	N/A	1	1	N/A
Character n-gram size	N/A	N/A	3-6	N/A
Epochs	N/A	10	10	N/A

5.3.3 Training Data

To train the taggers, I used the data extracted from the A Talking Dictionary of Ainu: A New Version of Kanazawa's Ainu Conversational Dictionary by Bugaeva et al. [21]. Apart from isolated headwords, the dictionary includes 2,459 multi-word items (phrases and sentences) and each of them is annotated with a sequence of POS tags. Using that information, I was able to build a small (12,952 token-tag pairs, excluding punctuation) part-of-speech annotated corpus. A subset of it was excluded from the training data, in order to be used as test data in the evaluation experiments (for details, see the next section), which left me with a training corpus of 11,249 token-tag pairs (excluding punctuation).

In order to avoid an increase in the number of out-of-vocabulary words, I decided to retain single-word entries in the training corpus and treated them as separate sentences (by inserting a sentence delimiter after each of them).

The corpus was prepared in column format (one token per line). Additionally, for the purpose of applying it with POST-AL, it was converted into a dictionary format, where each entry consists of a token (word or punctuation mark), part-of-speech and a list of sentences the given word appears in (if available). The resulting

dictionary contains a total of 2,392 entries.

5.3.4 Test Data

To evaluate the performance of the taggers, I used two sets of held-out data:

- **TDOA:** This dataset consists of 2,023 tokens (including 322 punctuation tokens) from the A Talking Dictionary of Ainu... [21]. Samples for the test data were selected in the following way: firstly, all sentences with the token count (excluding punctuation) of 3 and higher were extracted from the training corpus and grouped according to their token count. Secondly, duplicate sentences were eliminated. In the next step, a random sample of 20% was selected from each group. Lastly, the sentences selected for the test data were excluded from the training corpus.

- **SYOS:** Five out of thirteen yukar epics included in the *Ainu Shin'yōshū* ("Collection of Ainu songs of gods") by Yukie Chiri [29]. Unlike the A Talking Dictionary of Ainu..., it represents the literary style of Ainu. The text was revised in terms of transcription by Kirikae [65]. It comprises a total of 1,847 tokens (including 241 punctuation tokens) in 88 sentences. Of that number, 437 tokens (23.7%) do not appear in the training data. Part-of-speech annotations for this dataset were provided by professor Yoshio Momouchi (see Momouchi et al. [86]).

5.3.5 Part-of-speech Annotations and Tagset

Before applying the annotations produced by Bugaeva et al. in my experiments, I decided to introduce several modifications. All such decisions were consulted with three comprehensive dictionaries including the information about parts of speech, by Nakagawa [92], Tamura [143] and Kirikae [65]. I also referred to the classification of word classes proposed by Refsing [126].

The most notable change was the elimination of two word classes: Numeral (135 occurrences in the original data) and Interrogative (347 occurrences). All tags belonging to these two classes were converted to one of the following tags, depending on morpho-syntactic characteristics of words they denote: "Adnoun" (e.g.

sine – "one [day]") or "Noun" (e.g. *sinep* – "one thing") for Numerals, and "Pronoun" (e.g. *hemanta* – "what"), "Adnoun" (e.g. *inan* – "which"), "Noun" (*hempakniw* – "how many people"), "Adverb" (e.g. *hempara* – "when") or "Locative noun" (e.g. *hunak* – "where") for Interrogatives. The reason for that modification is that, apart from Bugaeva and Endō only Nakagawa classifies such words simply as Numerals and Interrogatives, whereas both Tamura and Kirikae rely on functional criteria in deciding their primary word class. Apart from that, I corrected a number inconsistent annotations and typos, and annotated words for which POS tags were missing in the original data.

Not all part-of-speech tags present in the out-of-domain data set (SYOS) appear in the A Talking Dictionary of Ainu. Most of them are punctuation marks, and thus are unambiguous. I decided to include three of them, which are common in other texts (namely, quotation mark – "", colon – ":" and ellipsis – "..."), in the training data. Specifically, I appended them at the end of the training samples, without any context.

The complete part-of-speech tagset along with statistics of occurrences in both datasets is presented in Table 5.35.

5.3.6 Results and Discussion

Results of POS tagging experiments using the SVMTool, for each combination of tagging parameters, are shown in Tables 5.38 and 5.39, while Table 5.37 presents the results of experiments with POST-AL. The results for the neural tagger (specifically, for the best training epoch of each model) are presented in Table 5.40. Table 5.36 shows the Most Frequent Tag baselines calculated by the SVMTool.

The results indicate that, while all taggers are better than the baseline, the models generated using the SVMTool and a neural tagger with pre-trained word embeddings perform better than POST-AL, especially when applied to out-of-domain data (SYOS). The biggest advantage of the two latter models is their ability to predict part-of-speech tags for out-of-vocabulary words (see Tables 5.41, 5.42 and 5.40), while POST-AL does not have such a mechanism. In fact, if OoV words were excluded from the calculation, in the experiment on the SYOS dataset POST-AL would rank first, with slightly higher accuracy than the SVMTool model (1,238 versus 1,234 correct predictions).

Table 5.35: Part-of-speech tagset and statistics.

Tag	Number of occurrences: A Talking Dictionary of Ainu... (with modifications)	Number of occurrences: SYOS
Noun	2,799	355
Intransitive verb	2,504	297
Transitive verb	1,503	174
Personal affix	1,114	178
Adverb	1,041	65
Conjunctional particle	626	146
Nominalizer	594	36
Locative noun	480	64
Final particle	430	22
Case particle	415	55
Adnoun	343	38
Postpositive adverb	246	8
Verb auxiliary	229	50
Supplementary particle	182	28
Pronoun	166	8
Ditransitive verb	130	18
Complete verb	56	2
Interjection	47	11
Proper noun	47	17
Prefix	0	3
.	3,396	55
;	508	0
?	470	12
,	106	102
!	28	14
"	1	50
:	1	1
...	1	2
:–	0	5
Unknown	0	31

Table 5.36: Most Frequent Tag baseline.

Test data	Accuracy
TDOA	1910 / 2023 (94.41%)
SYOS	1225 / 1847 (66.32%)

Table 5.37: Results of the experiments with POST-AL.

Test data	Accuracy
TDOA	1939 / 2023 (95.85%)
SYOS	1238 / 1847 (67.03%)

Differences in accuracy observed between various tagging strategies offered by the SVMTool were also mainly caused by different scores for unknown words, while the results for known words exhibited much less variance. For instance, in the experiment on SYOS, the performance for in-vocabulary words was less than 1% higher with the tagging strategy set to `-T 4` as compared to `-T 0` (1,234 versus 1,224 correct predictions), but at the same time the performance for OoV words improved by over 12% (213 versus 160 correct predictions).

The best performance with both sets of test data was achieved by tagging strategies 4 and 6. According to the SVMTool's technical manual [45], both of them utilize Model 4 — the variant which addresses the problem of OoV words by artificially marking a portion of the training data as unknown during the learning process. Additionally, tagging strategy 6 maximizes the global (sentence-level) sum

Table 5.38: Results (Accuracy) of the experiments with the SVMTool on TDOA (best result in bold).

		Direction (`-S`)		
		LR	**LRL**	**RL**
Tagging strategy (`-T`)	0	97.33%	97.08%	89.77%
	1	97.68%	96.89%	90.46%
	2	97.62%	97.23%	90.21%
	4	97.78%	97.03%	89.27%
	5	97.33%	96.89%	90.11%
	6	**97.83%**	96.79%	89.42%

Table 5.39: Results (Accuracy) of the experiments with the SVMTool on SYOS (best result in bold).

		Direction (- S)		
		LR	**LRL**	**RL**
Tagging strategy (- T)	0	74.93%	74.07%	69.95%
	1	77.31%	75.64%	69.46%
	2	76.99%	76.77%	72.93%
	4	**78.34%**	75.04%	69.84%
	5	75.09%	75.26%	70.06%
	6	78.07%	75.96%	69.90%

Table 5.40: Results (Accuracy) of the experiments using the neural tagger with different types of word representations (best results in bold).

Test data	**One-hot**	**word2vec**	**fastText**	**Cross-ling.**	**fastText +cross-ling.**
TDOA	97.63%	97.53%	97.68%	97.78%	**98.02%**
Known (1977)	98.48%	98.33%	98.38%	98.43%	**98.63%**
OoV (46)	60.87%	63.04%	67.39%	69.57%	**71.74%**
SYOS	72.44%	75.80%	77.10%	74.93%	**78.34%**
Known (1410)	82.62%	82.91%	83.48%	82.98%	**84.11%**
OoV (437)	39.59%	52.86%	56.52%	48.97%	**59.73%**

of SVM scores, rather than making decisions based on a reduced context.

Contrary to the results reported by Giménez and Márquez, using the combination of both tagging directions (- S LRL) did not improve the performance in my experiments – the only case where it yielded slightly higher accuracy than tagging from left to right (- S LR) was the experiment on SYOS with tagging strategy set to 5.

The proposed method for generating cross-lingual word embeddings resulted in considerably better performance than with one-hot encodings, but not as good on out-of-domain data as the state-of-the-art monolingual word embedding techniques (especially fastText). Nevertheless, in an additional experiment, where the proposed embeddings were combined (concatenated) with fastText-generated ones, it outperformed other models by a significant margin, yielding accuracies on par with the SVMTool.

Table 5.41: Results of the experiments with SVMTool (`- T 6 - S LR`) on TDOA – Accuracy per category of words.

Category	Accuracy
Known	1950 / 1977 (98.63%)
OoV	29 / 46 (63.04%)

Table 5.42: Results of the experiments with SVMTool (`- T 4 - S LR`) on SYOS – Accuracy per category of words.

Category	Accuracy
Known	1234 / 1410 (87.52%)
OoV	213 / 437 (48.74%)

As described in Section 5.3.5, some of the tags appearing in the out-of-domain data set (SYOS) were absent from the A Talking Dictionary of Ainu, and three of them (i.e. "!", ":" and "...") were artificially added to the training corpus. Experiments revealed that, while for POST-AL and the SVMTool a single annotated instance without context was enough to learn the correct tag for such unambiguous tokens, the neural model tagged them incorrectly. This problem could be easily solved by establishing heuristic rules for handling punctuation marks, or by expanding the training corpus with examples including such tokens and their context. To illustrate the effect of this issue on the experiment, I calculated the accuracy of each tagger on SYOS after excluding the six classes not seen in the A Talking Dictionary of Ainu...: "!", ":", "...", ":–", "Unknown" and "Prefix". The results are presented in Table 5.43. As expected, the biggest performance gains were obtained by the neural model, and this time the variants utilizing pre-trained word embeddings outperformed the SVMTool.

Table 5.43: Part-of-speech tagging accuracy on SYOS after excluding 6 word classes not seen in the A Talking Dictionary of Ainu... (best results in bold).

MFT	**POST-AL**	**SVMTool**	**One-hot**	**word2vec**	**fastText**	**Cross-ling.**	**fastText +cross-ling.**
66.78%	67.52%	79.43%	76.24%	79.77%	81.03%	78.86%	**82.39%**

5.3.7 Conclusions

In this research I used a small amount of part-of-speech annotated Ainu language textual data to train and compare three POS taggers: POST-AL – a system developed specifically for Ainu, based on contextual (n-gram) and statistical (Term Frequency) information derived from a lexicon, a tagger generated using the SVMTool – an off-the-shelf generator of sequential taggers based on Support Vector Machines, and a neural POS tagger equipped with four different types of word representations.

Experiments conducted on two different sets of objective data revealed that the two latter approaches are more effective than the lexicon-based tagger (POST-AL), especially when applied to out-of-domain data, the main reason for higher accuracy being their ability to predict part-of-speech tags for out-of-vocabulary words.

While being outperformed by state-of-the-art techniques for pre-training word embeddings, the proposed method for generating cross-lingual word representations contributed positively to tagging accuracy. Moreover, word representations obtained by combining the proposed approach with fastText embeddings yielded the best results.

5.4 First Step towards Robot-assisted Language Learning for Ainu

While multiple initiatives are being undertaken by the members of the Ainu community to preserve their mother tongue and promote it among the young generations, the number of speakers who possess the level of proficiency necessary to teach the language, is extremely small. As a consequence, access to Ainu language education is severely limited.

In recent years, Computer-Assisted Language Learning (CALL) and Robot-Assisted Language Learning (RALL) have been proposed as a way to support both native and foreign language acquisition [125]. I believe that they could also be helpful in addressing the challenges facing Ainu language teaching.

A major obstacle for the application of such ideas to minority languages, including Ainu, is the lack of high-volume linguistic resources (such as text and speech corpora and in particular, annotated corpora) necessary for the development of dedicated text and speech processing technologies. Advances in cross-lingual learning indicate that this problem can be, to a certain extent, alleviated by transferring knowledge from resource-rich languages. Unsurprisingly, however, such techniques tend to yield the best results for closely related languages (see, e.g., [49]), which is not a feasible scenario for the Ainu language, as it has no known cognates. Nevertheless, given the similarity of phonological systems and some (presumably, contact-induced) grammatical constructions between Ainu and Japanese, I anticipate that it may be beneficial to use the existing Japanese resources as a starting point in the development of language processing technologies for Ainu.

In this section, I describe a preliminary Ainu language conversational program for the Pepper robot[15], which serves as a proof of concept of how robots could support Ainu language education. Because dedicated speech technologies for Ainu are not available[16], and to test my assumptions about potential benefits of Japanese-Ainu cross-lingual transfer, I use Text-to-Speech and Speech Recognition models for Japanese. Finally, I conduct a survey among a group of Ainu language experts and experienced learners to evaluate the robot's speech in terms of intelligibil-

[15] https://www.softbankrobotics.com/emea/en/pepper

[16] In a recent work, Matsuura et al. [80] reported developing an end-to-end Speech Recognition model for Ainu, but as of today, their system is not publicly available.

ity and pronunciation, and perform automatic evaluation of Speech Recognition performance.

5.4.1 Materials

5.4.1.1 Pepper Robot

Pepper (shown in Figure 5.1) is a humanoid robot manufactured by SoftBank Robotics. It was first introduced in 2014. Below, I provide a short description of the robot's three components relevant to my experiments: Speech Synthesis (Text-to-Speech), Speech Recognition and dialogue scripting functionality.

Speech Synthesis Conversion of Japanese text input to speech is performed by the microAITalk engine[17]. It uses a concatenative speech synthesizer based on phonemic and prosodic patterns learned from a speech corpus[18]. Speech parameters, such as pitch, speed and volume, can be locally adjusted by inserting special tags in the text.

Speech Recognition Pepper is equipped with a closed-vocabulary Speech Recognition engine, i.e., a list of possible words/phrases must be predefined. Once speech has been detected, each phrase from the list is assigned a confidence level ("estimate of the probability that this phrase is indeed what has been pronounced by the human speaker")[19].

Dialogue Scripting A dedicated language, QiChat, is used to define "rules" for managing the flow of the conversation between the robot and humans[20]. There are two types of rules: "User rules", triggered by human input, and "Proposal rules" generating specific robot output without any user input. Rules are grouped by "Topics". Apart from rules, a Topic may include "Concepts" (lists of equivalent or related words/phrases, e.g., synonyms of a word), as well as user-defined functions.

[17] http://doc.aldebaran.com/2-5/naoqi/audio/altexttospeech.html
[18] https://www.ai-j.jp/about/
[19] http://doc.aldebaran.com/2-5/naoqi/audio/alspeechrecognition.html
[20] http://doc.aldebaran.com/2-5/naoqi/interaction/dialog/aldialog.html

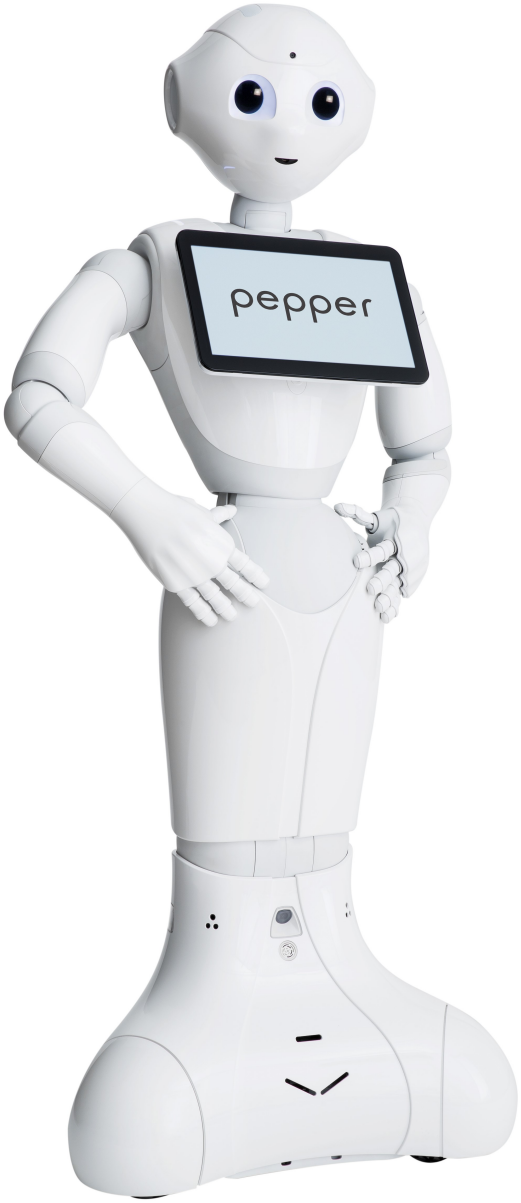

Figure 5.1: "Pepper the Robot", by Softbank Robotics Europe. Source: https://commons.wikimedia.org/wiki/File:Pepper_the_Robot.jpg. Licensed under the Creative Commons Attribution-ShareAlike 4.0 International license: https://creativecommons.org/licenses/by-sa/4.0/

5.4.1.2 Dialogue in the Ainu Language

Using Choregraphe[21] (a graphical environment for programming Pepper) and QiChat, I created a simple dialogue script in the Ainu language. The script includes 78 User rules and subrules, 20 Proposal rules, 93 Concepts and 2 function definitions. Two excerpts from the script (four Concept definitions and a single User rule with subrules) are shown in Listing 5.2. Possible conversation topics include: greetings, self-introduction (name, age, etc.) and asking about weather. If the human interlocutor does not speak for a specified period of time, the robot will initiate conversation by asking a question. The user can also ask to translate words (included in a predefined list) from Japanese to Ainu. Furthermore, when Pepper asks a question in the Ainu language, the user can ask him to repeat it or to explain its meaning in Japanese. In addition to conversations, the script includes two interactive games played in the Ainu language, of which one utilizes the robot's touch sensors.

Although the bulk of the dialogues in the script are in Ainu, the parameter determining which language should be used by the robot's Text-to-Speech engine and Speech Recognition engine, was set to Japanese. This effectively turned my experiment into an instance of cross-lingual knowledge transfer (in particular, the simplest form of it where a model trained solely on source language data is applied to the target language – see [49]). An additional benefit of using Japanese speech technologies is the ability to communicate with the robot not only in Ainu, but also in the first language of most learners of the Ainu language.

The contents of the dialogues were based mainly on materials for the Saru dialect of Ainu (spoken in southern Hokkaidō), such as [143, 60, 94] and textbooks for the "Ainugo Rajio Kōza" [Ainu Language Radio Course][22].

In cases where the intonation of utterances generated by the Speech Synthesis engine was remarkably inconsistent with the recordings by native speakers (e.g., questions pronounced by the robot with a falling intonation, whereas it should be rising), I used the tags mentioned in Section 5.4.1.1 to modify pitch and speed. However, in this preliminary experiment I refrained from performing extensive

[21] http://doc.aldebaran.com/2-5/software/choregraphe/
[22] https://www.stv.jp/radio/ainugo/index.html

Listing 5.2 Excerpts from the QiChat script

```
concept:(and_you) "\rspd=100\\vct=100\エアニ \rspd=95\\vct=120\ヘエエ?"
concept:(and_you2) ^rand["\rspd=100\\vct=100\エアニ カ エ \rspd=110\イワンケエ?" ~and_you] # Only in response to "how are you?"
concept:(me) "\rspd=100\\vct=100\クアニ"
concept:(me_too) "~me カ"
.....
u:({ペ ッ パ ー 、 } !ク エ イ ワ ン ケ {"ワ エ ア ン"} [ア ヤ]) "ク \rspd=110\\vct=100\イワンケ \rspd=100\\vct=135\ワアア". ~and_you2 # "How are you (Pepper)?" -> "I'm fine. And you?"
  u1:(~me_too クイワンケ {ワ} {クアン}) ~good_to_hear # "I'm fine, too" -> "Good to hear that"
  u1:({ソ ン ノ} ク シ ン キ {ワ}) "\rspd=100\\vct=100\ポ ン ノ シ ニ \rspd=100\\vct=135\ヤアン" # "I'm (really) tired" -> "Please get some rest"
  u1:(ク ミシム) ~in_this_case ~lets_play # "I'm bored" -> "Then let's play!"
```

fine-tuning[23]. Furthermore, in a number of sentences in audio materials used for reference, I observed a noticeable increase in the length of the sentence's final vowel. While vowel length is not a distinctive feature in Hokkaidō Ainu, and thus it is not reflected in written texts, I adjusted the transcriptions of such fragments in order to make the robot's pronunciation resemble that of native speakers. Examples include the sentence-final particle *yan*, used in polite commands: in the case of *katakana* transcription it is normally expressed as ヤン (/yan/), but in one of the sentences in my dialogue script (see Listing 5.2) it was instead transcribed as ヤアン (/yaan/). As a result, Pepper pronounced it as [jaːn].

Since the majority of syllable-final consonants are not supported by the Japanese speech models, the corresponding full-size *katakana* characters (representing open, CV syllables) were used to denote them. For example, the word *anakne* (/anakne/, "as for") was transcribed as アナクネ (/anakune/).

[23]This decision was in part motivated by the observation that insertion of multiple tags in a single word or short utterance can cause problems with the Speech Synthesis engine, leading to unnatural output and unintended pauses.

5.4.2 Evaluation Experiments

5.4.2.1 Survey

In the first experiment, I asked Ainu language experts and experienced learners for a judgement of the quality of speech generated by the robot, as well as for a feedback about the idea of creating an Ainu language-speaking robot and using it in language education.

Participants The survey was conducted among a group of 8 people engaged in activities related to the Ainu language, namely: 4 learners of Ainu, 1 language instructor, 1 linguist and 2 persons involved in other types of activities. 7 out of 8 participants have at least 5 years of experience with the Ainu language. All participants are also speakers of Japanese, including 7 native speakers.

Survey design The participants were asked to watch a video demonstrating a conversation with the robot[24] and evaluate the quality of its speech (both in Japanese and Ainu) in terms of intelligibility (defined as the listener's ability to identify the words spoken by the robot) and correctness of pronunciation. The latter was further broken up into four different aspects: pronunciation of Japanese/Ainu sounds, accents, intonation and overall impression. Intelligibility was evaluated on a three-point scale: "easy to understand", "sometimes hard to understand", "hard to understand". For pronunciation I employed a five-point scale: "perfect", "quite good", "some problems", "not good", "very bad" (when calculating average scores, presented later in this chapter, I converted each grade to a numerical value, where "perfect" corresponds to 5 and "very bad", to 1).

The final question asked for an opinion as to whether an Ainu language-speaking robot such as the one developed in this research could be useful in Ainu language education. Here, the predefined options were "yes", "yes, if it's improved" and "no"; the participants were also given an option to specify their own answer.

Apart from the closed questions, the respondents were encouraged to submit any additional opinions and comments.

[24] https://youtu.be/DJgVolvcees. A version with subtitles (in Ainu and Japanese) is also available: https://youtu.be/RTuGqgDNBC8.

5.4.2.2 Speech Recognition Experiment

The goal of the second experiment was to investigate the performance of the robot's Japanese Speech Recognition engine in identifying speech uttered in the Ainu language. For that purpose, I selected a list of 30 Ainu words – of which 12 include combinations of sounds violating phonotactic constraints observed in the Japanese language – and 30 Japanese words. Each of the two word lists was then presented to the Speech Recognition engine as the list of possible phrases to detect. All words were transcribed in *kana* script (namely, *hiragana* for Japanese and *katakana* for Ainu words). As in the dialogue script (see Section 5.4.1.2), consonants occurring in syllable coda position in Ainu words were represented with the corresponding standard *katakana* characters.

Finally, a recording of each word from the respective list, uttered by a native speaker, was played to the robot 3 times and the output was recorded. Japanese voice recordings were obtained from the WWWJDIC online dictionary[25]. In the case of Ainu words, I used the audio pronunciations recorded by Shigeru Kayano for his dictionary [60] and included in the online version released by the Ainu Museum[26].

Results of the experiment were evaluated in terms of Accuracy (defined as the proportion of trials where the target word yielded the highest probability) and confidence levels for words assigned the highest value in each trial.

5.4.3 Results and Discussion

As shown in Figure 5.2, more than half of the respondents did not have any problems with understanding Pepper's speech in Japanese. On the contrary, the speech in Ainu was at times difficult to understand for three quarters of the participants.

In their comments, multiple respondents indicated that the robot's speech was sometimes too fast, which rendered it not only unnatural, but also difficult to follow. In addition to that, several respondents pointed out problems with intonation (e.g., in questions), which in their opinion also harmed intelligibility.

[25] http://nihongo.monash.edu/cgi-bin/wwwjdic
[26] https://ainugo.ainu-museum.or.jp/

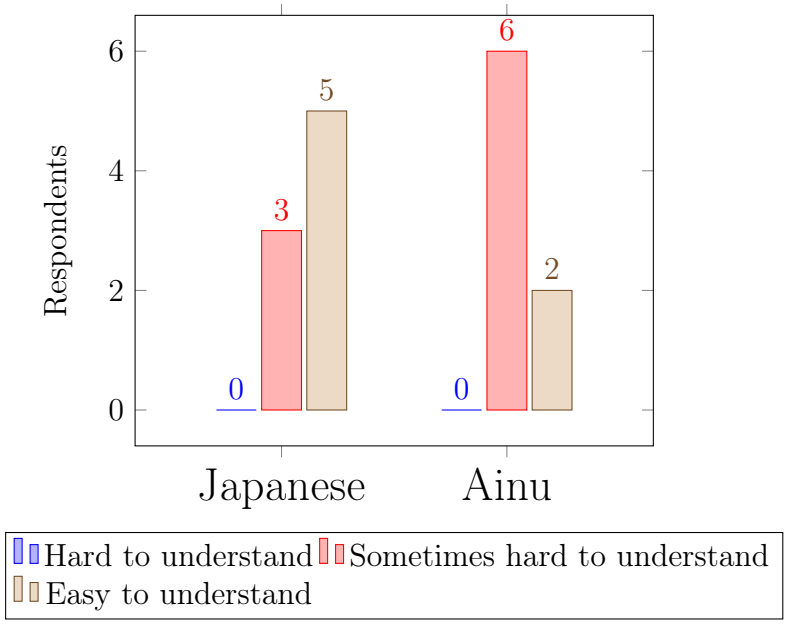

Figure 5.2: Evaluation of the robot's intelligibility

The majority of the respondents rated the robot's Japanese pronunciation as either "good" or "perfect" (Figure 5.3). In the case of Ainu pronunciation (Figure 5.4), all aspects received the middle grade ("some problems") or higher from more than half of the participants.

In both cases, the lowest average scores were achieved for accents (4.00 for Japanese and 3.00 for Ainu) and intonation (4.125 and 3.00). Relatively low results for Japanese can be attributed to the characteristics of the its pitch accent system (namely, unpredictability of the accent's location – see Section 2.2.1). This impairs the Text-to-Speech system's ability to generate correct prosodic patterns for words and phrases unseen in the training data. In Ainu, there is less uncertainty in terms of accent, but the Text-to-Speech model, trained exclusively on Japanese data, has no knowledge of the rules governing it.

The two participants with presumably the most relevant expertise (i.e., the person involved in linguistic research related to the Ainu language and the Ainu language instructor) were generally less favorable in their evaluations: on average they rated Pepper a 3.625 for Japanese speech and 2.25 for the pronunciation of Ainu, whereas the average for all respondents was 4.28 (for Japanese) and 3.16 (for Ainu). As for specific points of criticism, both of them reported hearing consonants

followed by vowels in places where only a consonant (belonging to the coda of the preceding syllable) should appear. That, of course, is the effect of restrictions on closed syllables in the Japanese Text-to-Speech engine.

No comments were made concerning pronunciation of individual phonemes of the Ainu language. This suggests that the similarities to Japanese in this area are significant enough that a Japanese Text-to-Speech model can produce reasonable results.

Figure 5.5 shows the results of the survey's final question. Apart from a single respondent who was skeptical about the willingness of the Ainu people to learn their mother tongue from a machine, all participants expressed an opinion that a robot of this type would be useful in Ainu language education, either now (50%) or after improvements (37.5%).

Speech Recognition experiment results are summarized in Table 5.44. The robot correctly selected the target word as the most probable option in all trials for both languages. Moreover, in both cases the average confidence level exceeded 50% – the default confidence threshold in the dialogue engine, below which the speech recognition result is ignored[27]. On the other hand, in the experiment with Japanese words, only one trial yielded a confidence value below the threshold, while for words in the Ainu language, the proportion of such results was 28%. Not surprisingly, Ainu words containing syllables violating Japanese phonotactic rules perplexed the Speech Recognition engine to a greater extent than the rest of them, achieving an average confidence level of 50.36%. The value for the subset of words with two such syllables was even lower: 44.88%.

5.4.4 Conclusions

This research is a first step in a project to develop an Ainu language-speaking robot, intended for use in language education. In order to demonstrate the concept to potential users, I utilized existing technologies (i.e., the Pepper robot) to create a rule-based dialogue agent capable of holding simple conversations, teaching new words and playing interactive games in Ainu. After presenting the robot to a group

[27] Although the threshold can be freely modified in Speech Recognition options, setting it to a low value may negatively affect precision of the Speech Recognition engine, causing it to recognize non-verbal sounds or background noise as speech.

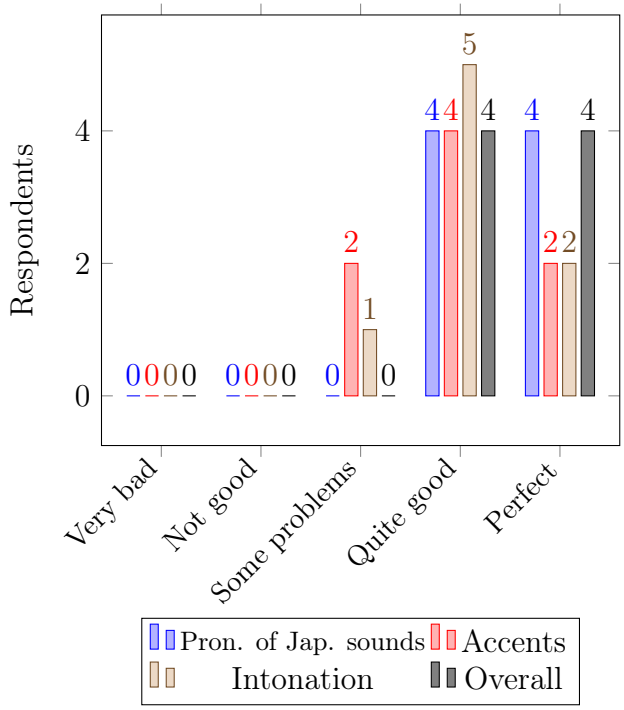

Figure 5.3: Evaluation of the robot's Japanese pronunciation

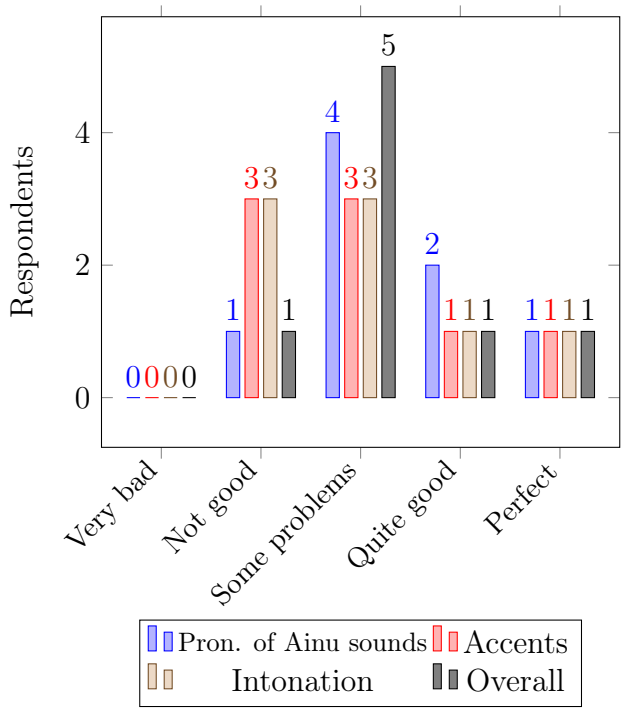

Figure 5.4: Evaluation of the robot's Ainu language pronunciation

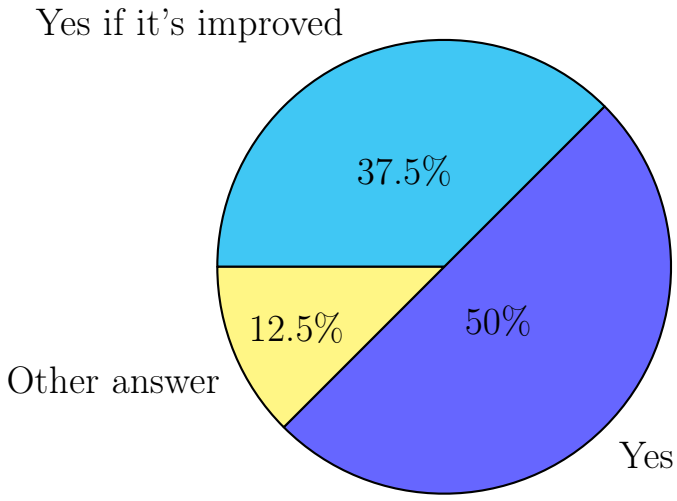

Figure 5.5: Answers to the question: "Do you think that an Ainu language-speaking robot such as the one in the video could be useful in Ainu language education?"

		Japanese	**Ainu**
	ACCURACY:	100.00%	100.00%
CONFIDENCE	**Minimum:**	45.90%	39.84%
	Maximum:	76.13%	66.47%
	Mean:	64.14%	54.29%
	Median:	64.21%	54.31%

Table 5.44: Speech Recognition experiment results

of Ainu language learners and experts (in the form of a demonstrational video), I received positive feedback about the idea.

Existing tools for dialogue scripting, such as QiChat, provide an intuitive way to manage closed-domain conversations. I envision that, once a general framework for robot-assisted Ainu language teaching is established, language experts and instructors with basic training in computer programming could easily expand its knowledge base by designing new rules and topics. In the future, I will also engage in the development of an AI-based, open-domain dialogue agent, or a hybrid system combining Artificial Intelligence with a rule-based approach.

At present, there exist no robots with Speech Synthesis and/or Speech Recognition technologies supporting the Ainu language. Leveraging similarities between

phonological systems of Ainu and Japanese, in this preliminary work I utilized the Japanese models supplied with Pepper. Evaluation of the robot-generated speech by a group of experts and experienced learners revealed that – due to differences in phonotactics and suprasegmental features – employing a Japanese Text-to-Speech model alone is not sufficient to produce high-quality output (especially if one intends to use the system in language teaching). That being said, the results of both evaluation experiments seem to confirm the potential of cross-lingual transfer from Japanese for facilitating the development of dedicated speech technologies in the low-resource setting of Ainu, in particular in the context of Speech Recognition. I believe that the insights from this research will be useful in that process.

Chapter 6

Conclusions and Future Work

6.1 Summary

In this thesis, I presented the results of my research devoted to the development of Natural Language Processing technologies for the Ainu language, a critically endangered language isolate spoken in the northern parts of the Japanese archipelago.

The main contributions of this research can be summarized as follows:

1. A roadmap for the development of Natural Language Processing (NLP) Technologies for Ainu, based on empirical study of actual needs of the potential users.

2. Compilation of a unified corpus of the Ainu language and positive verification of its utility in Natural Language Processing tasks.

3. Application of state-of-the-art Natural Language Processing technologies to Ainu, and development of dedicated technologies.

4. Improvements in three main NLP tasks for Ainu: transcription normalization, word segmentation and part-of-speech (POS) tagging.

5. Proposal of a novel type of word segmentation model: MiNgMatch.

I began this thesis with a description of the characteristics and current situation of the Ainu language, along with an overview of previous research, including the

few existing studies in the field of Natural Language Processing (Chapter 2).

In Chapter 3, I reported the results of a questionnaire-based empirical study, which confirmed the demand for language-related technologies for the Ainu language. Based on the findings from the survey, I outlined a general framework for Ainu language research involving Natural Language Processing techniques.

In Chapter 4, I described the scope and structure of the the first large-scale digital corpus of the Ainu language.

In Chapter 5, I presented the Natural Language Processing tools developed (or adapted to the Ainu language) in the course of this research. Firstly, in Section 5.1, I described the improvements introduced to POST-AL, a dictionary-based NLP toolkit originally developed by Ptaszynski and Momouchi [119]. In addition to improving the algorithms for transcription normalization, word segmentation, and part-of-speech tagging, I also expanded the system's dictionary base by combining two comprehensive Ainu language dictionaries. I found out that the combination improved overall performance of the tools, especially with objective samples unrelated to the training data.

After that, I reported the results of applying corpus-driven techniques (leveraging the Ainu language corpus presented in Chapter 4) to the tasks of word segmentation and part-of-speech tagging. Specifically, in Section 5.2 I introduced the MiNgMatch Segmenter: a data-driven word segmentation algorithm finding the minimal sequence of n-grams needed to match the input text. I compared my algorithm with segmenters utilizing two state-of-the-art n-gram language modelling techniques (namely, the Stupid Backoff model and a model with modified Kneser-Ney smoothing), as well as a neural model performing word segmentation as character sequence labelling.

The evaluation experiments revealed that the proposed approach is capable of achieving overall results comparable with the other best-performing models, especially when one takes into account the variance in notation of certain lexical items by different contemporary experts. Given its low computational cost and competitive results, I believe that MiNgMatch could be applied to other languages, and possibly to other Natural Language Processing problems, such as speech recognition.

In terms of precision of the segmentation process and accuracy for out-of-

vocabulary words, the sequence labelling neural model turned out to be the best option. In order to achieve that, however, it needs to be presented with training data tailored to the task, closely mimicking the intended target data. To this end, I demonstrated that such data can be bootstrapped from a small amount of manually annotated text, using the Universal Segmenter itself.

In Section 5.3, I described the results of an experiment with using a small amount of part-of-speech annotated Ainu language textual data to train and compare three POS taggers: POST-AL – a system developed specifically for Ainu, based on contextual (n-gram) and statistical (Term Frequency) information derived from a lexicon, a tagger generated using the SVMTool – a state-of-the-art generator of sequential taggers based on Support Vector Machines, and a neural POS tagger equipped with four different types of word representations: one-hot encodings, word embeddings generated with word2vec and fastText, and cross-lingual word representations derived from the Ainu-Japanese parallel corpus.

Experiments conducted on two different sets of objective data revealed that the two latter approaches are more effective than the lexicon-based tagger (POST-AL), especially when applied to out-of-domain data, the main reason for higher accuracy being their ability to predict part-of-speech tags for words unseen in the training data. The proposed method for building cross-lingual word representations contributed positively to the performance of the neural tagger, but was outperformed by state-of-the-art techniques for pre-training monolingual word embeddings (word2vec and fastText). On the other hand, the best results were achieved by combining the proposed representations with fastText embeddings.

In Section 5.4, I reported the results of a preliminary experiment in developing an Ainu language-speaking robot, intended for use in language education. In order to demonstrate the concept to potential users, I utilized existing technologies (i.e., the Pepper robot) to create a rule-based dialogue agent capable of holding simple conversations, teaching new words and playing interactive games in Ainu. After presenting the robot to a group of Ainu language learners and experts (in the form of a demonstrational video), I received positive feedback about the idea.

At present, there exist no robots with Speech Synthesis and/or Speech Recognition technologies supporting the Ainu language. Leveraging similarities between phonological systems of Ainu and Japanese, in this preliminary work I utilized the

Japanese models supplied with Pepper. Evaluation of the robot-generated speech by a group of experts and experienced learners revealed that – due to differences in phonotactics and suprasegmental features – employing a Japanese Text-to-Speech model alone is not sufficient to produce high-quality output (especially if one intends to use the system in language teaching). That being said, the results of both evaluation experiments seem to confirm the potential of cross-lingual transfer from Japanese for facilitating the development of dedicated speech technologies in the low-resource setting of Ainu, in particular in the context of Speech Recognition.

6.2 Future Directions

In the near future I will expand the corpus described in Chapter 4 with newly digitized materials. Another important task is the generation of part-of-speech annotations for all documents in the corpus. While the corpus-driven techniques investigated in this research allowed for significant improvements in tagging accuracy, in experiments on out-of-domain data none of the systems achieved human-level accuracy (see Section 5.3.6). To bridge the gap, I will convert other existing Ainu language resources including the information about parts of speech (such as the dictionary by Kirikae [65]) to a corpus format which could be used to train part-of-speech taggers. Another potential solution is to produce the necessary amount of training examples in an Active Learning scheme [136]. The part-of-speech annotated corpus will then be utilized in the development of further NLP tools, such as a morphological analyzer, a syntactic parser, speech technologies, and machine translation. In developing those technologies, I will further experiment with transferring linguistic knowledge from Japanese, and possibly also from other languages.

Future tasks concerning word segmentation include performing experiments with the MiNgMatch Segmenter on other languages and implementing an ensemble segmenter combining an n-gram model (such as MiNgMatch) with a neural model performing word segmentation as character sequence labelling. Another area that requires improvement is the handling of OoV words. All lexical n-gram-based models applied in the experiments performed poorly in this aspect and the proposed algorithm was not an exception. One possible way to increase the MiNgMatch

Segmenter's robustness against unseen forms might be to utilize character n-grams instead of word n-grams.

Appendix A

Feature Set Used in Experiments with the SVMTool

Tables A.1–A.4 present the feature sets defined for each of the four variants (Model 0/1/2/4) of the part-of-speech tagging model created in this research. Each of the tagging strategies offered by the SVMTool utilizes different variant(s) of the tagging model. For details, please refer to Giménez and Márquez [45].

Table A.1: Feature definition for Model 0.

	Feature category	**Definition**
	Word features	w_{-2}, w_{-1}, w_0, w_1, w_2
	POS features	p_{-2}, p_{-1}
	Ambiguity classes	a_0, a_1, a_2
	Maybe's	m_0, m_1, m_2
	Word bigrams	(w_{-2}, w_{-1}), (w_{-1}, w_0), (w_0, w_1), (w_{-1}, w_1), (w_1, w_2)
	POS bigrams	(p_{-2}, p_{-1}), (p_{-1}, p_1), (p_1, p_2)
	Word trigrams	(w_{-2}, w_{-1}, w_0), (w_{-2}, w_{-1}, w_1), (w_{-1}, w_0, w_1), (w_{-1}, w_1, w_2), (w_0, w_1, w_2)
	POS trigrams	(p_{-2}, p_{-1}, p_1), (p_{-1}, p_1, p_2)
Only for OoV words	Single characters	ca(1), cz(1)
	Prefixes	a(2), a(3), a(4)
	Suffixes	z(2), z(3), z(4)
	Lexicalized features	L (word length), SA (initial upper case), AA (all upper case), SN (starts with number), CA (any capital letter), CAA (several capital letters), CP (contains a period), CC (contains a comma), CN (contains a number), MW (contains a hyphen)

Table A.2: Feature definition for Model 1.

	Feature category	Definition
	Word features	w_{-2}, w_{-1}, w_0, w_1, w_2
	POS features	p_{-2}, p_{-1}, p_1, p_2
	Ambiguity classes	a_0, a_1, a_2
	Maybe's	m_0, m_1, m_2
	Word bigrams	(w_{-2}, w_{-1}), (w_{-1}, w_0), (w_0, w_1), (w_{-1}, w_1), (w_1, w_2)
	POS bigrams	(p_{-2}, p_{-1}), (p_{-1}, p_0), (p_{-1}, p_1), (p_0, p_1), (p_1, p_2)
	Word trigrams	(w_{-2}, w_{-1}, w_0), (w_{-2}, w_{-1}, w_1), (w_{-1}, w_0, w_1), (w_{-1}, w_1, w_2), (w_0, w_1, w_2)
	POS trigrams	(p_{-2}, p_{-1}, p_0), (p_{-2}, p_{-1}, p_1), (p_{-1}, p_0, p_1), (p_{-1}, p_1, p_2)
Only for OoV words	Prefixes	a(1), a(2), a(3), a(4)
	Suffixes	z(1), z(2), z(3), z(4)
	Lexicalized features	L, SA, AA, SN, CA, CAA, CP, CC, CN, MW

Table A.3: Feature definition for Model 2.

	Feature category	Definition
	Word features	w_{-2}, w_{-1}, w_0, w_1, w_2
	POS features	p_{-2}, p_{-1}
	Ambiguity classes	a_0
	Maybe's	m_0
	Word bigrams	(w_{-2}, w_{-1}), (w_{-1}, w_0), (w_0, w_1), (w_{-1}, w_1), (w_1, w_2)
	POS bigrams	(p_{-2}, p_{-1})
	Word trigrams	(w_{-2}, w_{-1}, w_0), (w_{-2}, w_{-1}, w_1), (w_{-1}, w_0, w_1), (w_{-1}, w_1, w_2), (w_0, w_1, w_2)
Only for OoV words	Prefixes	a(1), a(2), a(3), a(4)
	Suffixes	z(1), z(2), z(3), z(4)
	Lexicalized features	L, SA, AA, SN, CA, CAA, CP, CC, CN, MW

Table A.4: Feature definition for Model 4.

	Feature category	**Definition**
	Word features	w_{-2}, w_{-1}, w_0, w_1, w_2
	POS features	p_{-2}, p_{-1}
	Ambiguity classes	a_0, a_1, a_2
	Maybe's	m_0, m_1, m_2
	Word bigrams	(w_{-2}, w_{-1}), (w_{-1}, w_0), (w_0, w_1), (w_{-1}, w_1), (w_1, w_2)
	POS bigrams	(p_{-2}, p_{-1}), (p_{-1}, p_1), (p_1, p_2)
	Word trigrams	(w_{-2}, w_{-1}, w_0), (w_{-2}, w_{-1}, w_1), (w_{-1}, w_0, w_1), (w_{-1}, w_1, w_2), (w_0, w_1, w_2)
	POS trigrams	(p_{-2}, p_{-1}, p_0), (p_{-2}, p_{-1}, p_1), (p_{-1}, p_0, p_1), (p_{-1}, p_1, p_2)
Only for OoV words	Prefixes	a(1), a(2), a(3), a(4)
	Suffixes	z(1), z(2), z(3), z(4)
	Lexicalized features	L, SA, AA, SN, CA, CAA, CP, CC, CN, MW

Appendix B

Source Code

Listing B.1 presents the implementation of the transcription normalization and word segmentation algorithms described in section 5.1. The code is written in Perl 5.

Listing B.1: Code of the transcription normalization and word segmentation algorithms described in 5.1

```
#!/usr/bin/perl -s

use utf8;
use List::MoreUtils qw(uniq);
use File::Glob ':bsd_glob';
use File::Glob ':glob';

binmode(STDOUT, ":utf8");
binmode(STDIN, ":utf8");

my @words;
if($dic) {
        open( my $filehandle, "<", $dic ) or die "No such
           ↪ file! $dic: $!";
        foreach my $entry (my @temp = <$filehandle>){
                utf8::decode($entry);
                chomp $entry;
                $entry = lc($entry);
```

```
                if ($entry =~ /<word>(.*?)<\/word>/)
                {
                        $word = $1;
                }
                push(@words, $word);
        }
}
else {
        print "Please␣provide␣path␣to␣a␣dictionary␣file.";
}

my @words_uniq = uniq @words;
@sorted = sort { length($b) <=> length($a) || $a cmp $b }
   ↪ @words_uniq;

# Define transcription normalization rules
my %rules = ( ch=>c, sh=>s, ai=>ay, ui=>uy, oi=>oy, ei=>ey,
   ↪ au=>aw, iu=>iw, eu=>ew, ou=>ow, mb=>np, mp=>np, b=>p, g
   ↪ =>k, d=>t );

while (<>) {
        utf8::decode($_);
        my $input = $_;
        chomp $input;
        $input = lc($input);
        @input = split(/ /, $input);
        my @out_tokens = ();

        foreach my $input_token (@input) {
                $input_token_orig = $input_token;
                @input_token_var = ($input_token);

                # Apply transcription normalization rules if
                   ↪ needed
                if($normalize==1) {
```

```
                $input_token_temp = $input_token;
                $input_token =~ s/(ch|sh|ai|ui|oi|ei|
                   ↪ mb|mp|b|g|d)/{$rules{$1},$1}/g;
                $input_token =~ s/(au|iu|eu|ou)/{$1,
                   ↪ $rules{$1}}/g;
                @input_token_var = < $input_token >;

                if ($input_token_temp =~ /shi/) {
                        $input_token_temp =~ s/shi/{
                           ↪ si,s,shi}/g;
                        $input_token_temp =~ s/(ch|sh
                           ↪ |ai|ui|oi|ei|mb|mp|b|g|
                           ↪ d)/{$rules{$1},$1}/g;
                        $input_token_temp =~ s/(au|iu
                           ↪ |eu|ou)/{$1,$rules{$1
                           ↪ }}/g;
                        my @input_token_var2 = <
                           ↪ $input_token_temp >;
                        push @input_token_var,
                           ↪ @input_token_var2;
                        @input_token_var = uniq
                           ↪ @input_token_var;
                }

                map { $_ =~ s/(^\s|\s$)//g; }
                   ↪ @input_token_var;
        }

        my @candidates = ();
        foreach my $token_var (@input_token_var) {
                my @candidates_temp = grep{
                   ↪ $token_var =~ /$_/ } @sorted;
                push @candidates, @candidates_temp;
                my $token_var_temp = $token_var;
                foreach my $known (@candidates_temp)
```

```
                ↪ {
                        $token_var_temp =~ s/$known/
                            ↪ /g;
                }
                $token_var_temp =~ s/(^\s+|\s+$)//g;
                my @unknown = split(/ /,
                    ↪ $token_var_temp);
                push @candidates, @unknown;
        }
        @candidates = uniq @candidates;

        if($normalize==1) {
                my @temp = grep(/(ch|sh|ai|ui|oi|ei|
                    ↪ au|iu|eu|ou|mb|mp|b|g|d)/,
                    ↪ @candidates);
                foreach my $temp (@temp) {
                        my $temp_mod = $temp;
                        $temp_mod =~ s/ch/c/g;
                        $temp_mod =~ s/sh/s/g;
                        $temp_mod =~ s/ai/ay/g;
                        $temp_mod =~ s/ui/uy/g;
                        $temp_mod =~ s/oi/oy/g;
                        $temp_mod =~ s/ei/ey/g;
                        $temp_mod =~ s/au/aw/g;
                        $temp_mod =~ s/iu/iw/g;
                        $temp_mod =~ s/eu/ew/g;
                        $temp_mod =~ s/ou/ow/g;
                        $temp_mod =~ s/mb/np/g;
                        $temp_mod =~ s/mp/np/g;
                        $temp_mod =~ s/b/p/g;
                        $temp_mod =~ s/g/k/g;
                        $temp_mod =~ s/d/t/g;
                        if (grep{$temp_mod =~ /^$_$/
                            ↪ } @candidates) {
                                @candidates = grep {
```

```
                                    ↪ $_ ne $temp }
                                    ↪ @candidates;
                        }
                }
        }

        @candidates = grep { $_ ne '.' } @candidates;
           ↪  # Removes period (.) from the list of
           ↪ words to be matched (otherwise it would
           ↪  match any character)

        my $re1 = join '|', sort{ length( $b ) <=>
           ↪ length( $a ) }@candidates;

        my @matched = ();
        my $x = 1;
        my $y = length($input_token);

        do {
                my $re2 = "($re1)" x $x;

                foreach my $token_var (
                   ↪ @input_token_var) {
                        if ($#matched < 0) {
                                push @matched, grep
                                   ↪ defined(),
                                   ↪ $token_var =~
                                   ↪ /^$re2$/;
                        }
                }
                ++$x;
        }
        while ($#matched < 0 && $x < $y );

        if($#matched < 0) {
```

```
                        push @out_tokens, $input_token_orig;
                }
                else {
                        push @out_tokens, @matched;
                }
        }

        my $output = join("␣", @out_tokens);
        $output =~ s/\s+/ /g;
        $output =~ s/(^\s|\s$)//g;
        print "$output\n";
}
```

Listing B.2 presents the code of the neural part-of-speech tagger applied in experiments described in 5.3 (the variant using cross-lingual word representations). The code is implemented in Python 3.

Listing B.2: Neural part-of-speech tagger with cross-lingual word embeddings

```
"""
A simple neural POS tagger using cross-lingual
word representations derived from word alignments
with POS tags on the source language side.

Parts of the code are based on this tutorial:
https://blog.cambridgespark.com/tutorial-build-your-own-
    ↪ embedding-and-use-it-in-a-neural-network-e9cde4a81296
"""

import argparse
import tensorflow as tf
from tensorflow.keras.layers import Dense, Embedding,
    ↪ Activation, Flatten
from tensorflow.keras import Sequential
from tensorflow.keras.utils import to_categorical
import numpy as np
import collections
import csv

UNK_INDEX = 0
UNK_TOKEN = "UNK"

EOS_INDEX = 1
EOS_TOKEN = "EOS"

CONTEXT_SIZE = 2

# hyperparameters
HIDDEN_SIZE = 50
```

```
BATCH_SIZE = 128
EPOCHS = 10

def main():
        args = initialize()
        train_words = load_tagged_data(args.trainset)
        test_words = load_tagged_data(args.testset)
        oovs = get_oovs(train_words, test_words)
        # load word embeddings
        word_vectors, emb_dim = load_jpnPOS_vectors(args.
           ↪ embeddings)
        embedding_matrix = np.array(list(word_vectors.values
           ↪ ()))
        # map words and tags to integers
        word2id = {w: i for i, w in enumerate(word_vectors.
           ↪ keys())}
        tag2id, id2tag = get_tag_vocabulary(train_words)
        # add a vector for unknown words (a mean of all other
           ↪  embeddings)
        unk_vector = embedding_matrix.mean(0)
        embedding_matrix, word2id = add_new_word(UNK_TOKEN,
           ↪ unk_vector, UNK_INDEX, embedding_matrix,
           ↪ word2id)
        # add a random end-of-sequence vector
        eos_vector = np.random.standard_normal(emb_dim)
        embedding_matrix, word2id = add_new_word(EOS_TOKEN,
           ↪ eos_vector, EOS_INDEX, embedding_matrix,
           ↪ word2id)
        # convert data to integers
        X_train, Y_train = get_int_data(train_words, word2id,
           ↪  tag2id)
        X_test, Y_test = get_int_data(test_words, word2id,
           ↪ tag2id)
        # one-hot vector to encode the tag indexes
        Y_train, Y_test = to_categorical(Y_train),
```

```
            ↪ to_categorical(Y_test)
        # create the model
        pos_model = define_model(embedding_matrix, emb_dim,
            ↪ len(tag2id))
        pos_model.summary()
        id2word = sorted(word2id, key=word2id.get)
        # train and evaluate the model
        best_acc = 0
        eval_rslts = []
        best_epoch = 1
        for i in range(EPOCHS):
                print("Epoch␣{}/{}".format(i+1, EPOCHS))
                pos_model.fit(X_train, Y_train, batch_size=
                    ↪ BATCH_SIZE, epochs=1, verbose=1)
                acc = evaluate_epoch(pos_model, id2word,
                    ↪ X_test, Y_test)
                if acc > best_acc:
                        best_acc = acc
                        best_epoch = i+1
                        eval_rslts = evaluate_model(pos_model
                            ↪ , id2word, id2tag, X_test,
                            ↪ Y_test, oovs)
        print("Best␣epoch:␣{}".format(best_epoch))
        print("Accuracy:␣{:.2%}".format(best_acc))
        print()
        for ln in eval_rslts:
                print(ln)

def initialize():
        parser = argparse.ArgumentParser()
        parser.add_argument('-t', '--trainset', default=None,
            ↪  help='Path␣to␣training␣data')
        parser.add_argument('-v', '--testset', default=None,
            ↪ help='Path␣to␣test␣data')
        parser.add_argument('-e', '--embeddings', default=
```

```
            ↪ None, help='Path␣to␣embeddings')
        args = parser.parse_args()
        return args

def load_tagged_data(filepath):
        """
        Generates (word, pos) tuples.
        """
        tagged_words = []
        for line in read_lines(filepath):
                word = line.split()[0]
                tag = line.split()[1]
                tagged_words.append((word, tag))
        return tagged_words

def read_lines(filename):
        with open(filename, 'r', encoding='utf-8') as file:
                return [line.strip('\r\n') for line in file]

def load_jpnPOS_vectors(path):
        """
        Loads word vectors from a CSV file.
        """
        csv_data = csv.reader(read_lines(path), delimiter=','
           ↪ , quotechar='"')
        next(csv_data, None)  # skip the header line
        vectors = {}
        for row in csv_data:
                vectors[row[0]] = np.array([float(val) for
                   ↪ val in row[1:]])
        # get embedding dimension (= POS tagset size)
        emb_dim = len(row[1:])
        return vectors, emb_dim

def get_oovs(trainset, testset):
```

```
        """
        Finds out-of-vocabulary words in the test data.
        """
        oovs = collections.Counter()
        train_words = [item[0] for item in trainset]
        for item in testset:
                if item[0] not in train_words:
                        oovs[item[0]] += 1
        return oovs

def get_tag_vocabulary(tagged_words):
        """
        Accepts text in the form of (word, pos) tuples and
        returns a dictionary mapping POS tags to unique IDs.
        """
        tag2id = {}
        id2tag = {}
        for item in tagged_words:
                tag = item[1]
                tag2id.setdefault(tag, len(tag2id))
        for tag in tag2id:
                id2tag[tag2id[tag]] = tag
        return tag2id, id2tag

def get_int_data(tagged_words, word2id, tag2id):
        """
        Converts labelled words to arrays of integers (IDs).
        """
        X, Y = [], []
        span = 2 * CONTEXT_SIZE + 1
        buffer = collections.deque(maxlen=span)
        padding = [(EOS_TOKEN, None)] * CONTEXT_SIZE
        buffer += padding + tagged_words[:CONTEXT_SIZE]
        for item in (tagged_words[CONTEXT_SIZE:] + padding):
                buffer.append(item)
```

```
                window_ids = np.array([word2id.get(word) if (
                    ↪ word in word2id) else UNK_INDEX for (
                    ↪ word, _) in buffer])
                X.append(window_ids)
                middle_word, middle_tag = buffer[CONTEXT_SIZE
                    ↪ ]
                Y.append(tag2id.get(middle_tag))
        return np.array(X), np.array(Y)

def add_new_word(new_word, new_vector, new_index,
    ↪ embedding_matrix, word2id):
        """
        Adds a new word to the embedding matrix.
        """
        embedding_matrix = np.insert(embedding_matrix, [
            ↪ new_index], [new_vector], axis=0)
        word2id = {word: (index+1) if index >= new_index else
            ↪  index for word, index in word2id.items()}
        word2id[new_word] = new_index
        return embedding_matrix, word2id

def define_model(embedding_matrix, emb_dim, class_count):
        """
        Creates a simple part-of-speech model taking
        as input a tagged word and its context.
        """
        vocab_length = len(embedding_matrix)
        total_span = CONTEXT_SIZE * 2 + 1
        model = Sequential()
        model.add(Embedding(input_dim=vocab_length,
            ↪ output_dim=emb_dim, weights=[embedding_matrix],
            ↪  input_length=total_span))
        model.add(Flatten())
        model.add(Dense(HIDDEN_SIZE))
        model.add(Activation("tanh"))
```

```
        model.add(Dense(class_count))
        model.add(Activation("softmax"))
        model.compile(optimizer=tf.train.AdamOptimizer(),
           ↪ loss="categorical_crossentropy", metrics=["
           ↪ accuracy"])
        return model

def evaluate_epoch(model, id2word, x_test, y_test):
        """
        Calculate the accuracy of the model's predictions
           ↪ after the given epoch.
        """
        y_pred = model.predict_classes(x_test)
        error_counter = collections.Counter()
        for i in range(len(x_test)):
                correct_tag_id = np.argmax(y_test[i]) # turn
                   ↪ one-hot vector to index
                if y_pred[i] != correct_tag_id:
                        word = id2word[x_test[i][CONTEXT_SIZE
                           ↪ ]]
                        error_counter[word] += 1
        correct = len(x_test) - sum(error_counter.values())
        acc = correct / len(x_test)
        print("Accuracy:␣{}␣/␣{}␣({:.2%})".format(correct,
           ↪ len(x_test), acc))
        print()
        return acc

def evaluate_model(model, id2word, id2tag, x_test, y_test,
   ↪ oovs):
        """
        Calculate the accuracy of the model's predictions.
        """
        oovs_total = sum(oovs.values())
        in_vocab_total = len(x_test) - oovs_total
```

```
y_pred = model.predict_classes(x_test)
tag_counter = collections.Counter()
errors_per_class = collections.Counter()
errors_in_vocab = 0
errors_oovs = 0
for i in range(len(x_test)):
        correct_tag_id = np.argmax(y_test[i])
        tag_counter[correct_tag_id] += 1
        if y_pred[i] != correct_tag_id:
                word = id2word[x_test[i][CONTEXT_SIZE
                   ↪ ]]
                if word in oovs:
                        errors_oovs += 1
                else:
                        errors_in_vocab += 1
                errors_per_class[correct_tag_id] += 1
correct_in_vocab = in_vocab_total - errors_in_vocab
acc_in_vocab = correct_in_vocab / in_vocab_total
correct_oovs = oovs_total - errors_oovs
acc_oovs = correct_oovs / oovs_total
eval_rslts = ["Accuracy␣for␣known␣words:␣{}␣/␣{}␣
   ↪ ({:.2%})".format(correct_in_vocab,
   ↪ in_vocab_total, acc_in_vocab),]
eval_rslts.append("␣␣Accuracy␣for␣OoV␣words:␣{}␣/␣{}␣
   ↪ ({:.2%})".format(correct_oovs, oovs_total,
   ↪ acc_oovs))
eval_rslts.append("Results␣per␣word␣class:")
eval_rslts.append("TAG\tCORRECT\tTOTAL\tACCURACY")
for tag_id in tag_counter:
        correct = tag_counter[tag_id] -
           ↪ errors_per_class[tag_id]
        acc = correct / tag_counter[tag_id]
        eval_rslts.append("{}\t{}\t{}\t{:.2%}".format
           ↪ (id2tag[tag_id], correct, tag_counter[
           ↪ tag_id], acc))
```

```
        return eval_rslts

main()
```

Bibliography

[1] S. P. Abney and S. Bird. Towards a Data Model for the Universal Corpus. In *BUCC@ACL*, 2011.

[2] S. Abney and S. Bird. The Human Language Project: Building a Universal Corpus of the World's Languages. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 88–97, Uppsala, Sweden. Association for Computational Linguistics, July 2010.

[3] A. Anastasopoulos, M. Lekakou, J. Quer, E. Zimianiti, J. DeBenedetto, and D. Chiang. Part-of-Speech Tagging on an Endangered Language: a Parallel Griko-Italian Resource. *CoRR*, abs/1806.03757, 2018. arXiv: 1806.03757. URL: http://arxiv.org/abs/1806.03757.

[4] T. Arkhangelskiy, O. Belyaev, and A. Vydrin. The Creation of Large-Scale Annotated Corpora of Minority Languages using UniParser and the EANC platform. In *Proceedings of COLING 2012: Posters*, pages 83–92, Mumbai, India. The COLING 2012 Organizing Committee, Dec. 2012. URL: https://www.aclweb.org/anthology/C12-2009.

[5] Y. Azumi and Y. Momouchi. Development of analysis tool for hierarchical Ainu-Japanese translation data. *Bulletin of the Faculty of Engineering at Hokkai-Gakuen University*, 36:175–193, 2009.

[6] Y. Azumi and Y. Momouchi. Development of tools for retrieving and analyzing Ainu-Japanese translation data and their applications to Ainu-Japanese machine translation system. *Engineering Research: The Bulletin of Graduate School of Engineering at Hokkai-Gakuen University*, 9:37–58, 2009.

[7] J. Batchelor. *An Ainu–English–Japanese dictionary*. Hokkaidō-chō, 1889.

[8] P. Behera, A. K. Ojha, and G. N. Jha. Issues and Challenges in Developing Statistical POS Taggers for Sambalpuri. In Z. Vetulani, J. Mariani, and M. Kubis, editors, *Human Language Technology. Challenges for Computer Science and Linguistics*, pages 393–406, Cham. Springer International Publishing, 2018. ISBN: 978-3-319-93782-3.

[9] K. C. Berger, A. G. Hernaiz, P. Baroni, D. Hicks, E. Kruse, V. Quochi, I. Russo, T. Salonen, A. Sarhimaa, and C. Soria. *Digital Language Survival Kit: The DLDP Recommendations to Improve Digital Vitality*. 2018. URL: http://www.dldp.eu/sites/default/files/documents/DLDP_Digital-Language-Survival-Kit.pdf (visited on 10/10/2019).

[10] L. Besacier, E. Barnard, A. Karpov, and T. Schultz. Automatic speech recognition for under-resourced languages: A survey. *Speech Communication*, 56:85–100, 2014. ISSN: 0167-6393. URL: http://www.sciencedirect.com/science/article/pii/S0167639313000988.

[11] D. Biber. Representativeness in Corpus Design. *Literary and Linguistic Computing*, 8:243–257, 1993.

[12] S. Bird and D. Chiang. Machine Translation for Language Preservation. In *Proceedings of COLING 2012: Posters*, pages 125–134, Mumbai, India. The COLING 2012 Organizing Committee, Dec. 2012.

[13] A. Björkelund, A. Falenska, X. Yu, and J. Kuhn. IMS at the CoNLL 2017 UD Shared Task: CRFs and Perceptrons Meet Neural Networks. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 40–51, Vancouver, Canada. Association for Computational Linguistics, 2017.

[14] R. Blokland, M. Fedina, C. Gerstenberger, N. Partanen, M. Rießler, and J. D. Wilbur. Language Documentation meets Language Technology. In *Proceedings of 1st International Workshop in Computational Linguistics for Uralic Languages (IWCLUL 2015)*, 2015.

[15] R. Blokland, N. Partanen, M. Rießler, and J. Wilbur. Using Computational Approaches to Integrate Endangered Language Legacy Data into Documentation Corpora: Past Experiences and Challenges Ahead. In *Proceedings of the*

Workshop on Computational Methods for Endangered Languages, volume 2, 2019.

[16] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[17] T. Brants, A. C. Popat, P. Xu, F. J. Och, and J. Dean. Large Language Models in Machine Translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 858–867, Prague, Czech Republic. Association for Computational Linguistics, 2007. URL: https://www.aclweb.org/anthology/D07-1090.

[18] A. Bugaeva. Ainu complex predicates with reference to Japanese. In P. Pardeshi and T. Kageyama, editors, *Handbook of Japanese Contrastive Linguistics*, Handbooks of Japanese, pages 247–272. De Gruyter Mouton, 2018.

[19] A. Bugaeva. Internet applications for endangered languages: A talking dictionary of Ainu. *Waseda Institute for Advanced Study Research Bulletin*, 52(3):73–81, 2011.

[20] A. Bugaeva. Southern Hokkaido Ainu. In N. Tranter, editor, *The languages of Japan and Korea*, pages 461–509. Routledge, London, 2012.

[21] A. Bugaeva, S. Endō, S. Kurokawa, and D. Nathan. A Talking Dictionary of Ainu: A New Version of Kanazawa's Ainu Conversational Dictionary. A. Bugaeva and S. Endō, editors, 2010. URL: http://lah.soas.ac.uk/projects/ainu/ (visited on 11/25/2015).

[22] D. Cai and H. Zhao. Neural Word Segmentation Learning for Chinese. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 409–420, Berlin, Germany. Association for Computational Linguistics, 2016. URL: https://www.aclweb.org/anthology/P16-1039.

[23] S. F. Chen and J. Goodman. An Empirical Study of Smoothing Techniques for Language Modeling. In *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics*, ACL '96, pages 310–318, Santa Cruz, California. Association for Computational Linguistics, 1996.

[24] Chiba University Graduate School of Humanities and Social Sciences. Ainugo Mukawa Hōgen Nihongo – Ainugo Jiten [Japanese – Ainu Dictionary for the Mukawa Dialect of Ainu], 2014. URL: http://cas-chiba.net/Ainu-archives/index.html (visited on 02/25/2017).

[25] *Chikoro Utarpa ne Yesu Kiristo Ashiri Aeuitaknup Oma Kambi. The New Testament of our Lord and Saviour Jesus Christ in Ainu.* Translated by John Batchelor. Printed for the Bible Society's committee for Japan by the Yokohama bunsha, 1897.

[26] M. Chiri. *Bunrui Ainu-go jiten. Dai-ikkan: shokubutsu-hen* [Dictionary of Ainu, vol. I: Plants]. Nihon Jōmin Bunka Kenkyūsho, Tōkyō, 1953.

[27] M. Chiri. *Bunrui Ainu-go jiten. Dai-nikan: dōbutsu-hen* [Dictionary of Ainu, vol. II: Animals]. Nihon Jōmin Bunka Kenkyūsho, Tōkyō, 1962.

[28] M. Chiri. *Bunrui Ainu-go jiten. Dai-sankan: ningen-hen* [Dictionary of Ainu, vol. III: People]. Nihon Jōmin Bunka Kenkyūsho, Tōkyō, 1954.

[29] Y. Chiri. *Ainu shin-yōshū* [Collection of Ainu mythic epics]. Kyōdo Kenkyūsha, Tōkyō, 1923.

[30] S. Cooper, D. B. Jones, and D. Prys. Crowdsourcing the Paldaruo Speech Corpus of Welsh for Speech Technology. *Information*, 10:247, 2019. DOI: 10.3390/info10080247.

[31] M. Davies. Hansard Corpus. Part of the SAMUELS project, 2015. URL: https://www.english-corpora.org/hansard/ (visited on 05/25/2020).

[32] M. Dobrotvorskij. *Ainsko-russkij Slovar* [Ainu-Russian Dictionary]. Kazan, 1875.

[33] Y. Doval and C. Gómez-Rodríguez. Comparing Neural- and N-Gram-Based Language Models for Word Segmentation. *CoRR*, abs/1812.00815, 2018.

[34] H. Echizen'ya, K. Araki, and Y. Momouchi. Automatic extraction of bilingual word pairs using Local Focus-based Learning from an Ainu-Japanese parallel corpus. *Bulletin of the Faculty of Engineering at Hokkai-Gakuen University*, 32:41–63, 2005.

[35] G. Emerson, L. Tan, S. Fertmann, A. Palmer, and M. Regneri. SeedLing: Building and Using a Seed corpus for the Human Language Project. In *Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 77–85, Baltimore, Maryland, USA. Association for Computational Linguistics, June 2014. DOI: 10.3115/v1/W14-2211. URL: https://www.aclweb.org/anthology/W14-2211.

[36] S. Endō. Nabesawa Motozō ni yoru Ainugo no kana hyōki taikei: Kokuritsu Minzoku-gaku Hakubutsukan shozō hitsu-roku nōto kara [Ainu language notation method used by Motozō Nabesawa: from the written notes held by the National Museum of Ethnology]. *Kokuritsu Minzoku-gaku Hakubutsukan chōsa hōkoku* [Survey report of the National Museum of Ethnology], 134:41–66, 2016.

[37] M. Fang and T. Cohn. Learning when to trust distant supervision: An application to low-resource POS tagging using cross-lingual projection. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 178–186, Berlin, Germany. Association for Computational Linguistics, Aug. 2016. DOI: 10.18653/v1/K16-1018. URL: https://www.aclweb.org/anthology/K16-1018.

[38] M. Fang and T. Cohn. Model Transfer for Tagging Low-resource Languages using a Bilingual Dictionary. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 587–593, Vancouver, Canada. Association for Computational Linguistics, July 2017. DOI: 10.18653/v1/P17-2093. URL: https://www.aclweb.org/anthology/P17-2093.

[39] A. Fujii. Some notes on Japanese and English intonation. *Kagawa Daigaku Ippan Kyōiku Kenkyū*, 15:119–130, 1979. URL: http://shark.lib.kagawa-u.ac.jp/kuir/detail/3396201203270352 36?l=en.

[40] I. Fujimura, S. Chiba, and M. Ohso. Lexical and grammatical features of spoken and written Japanese in contrast: exploring a lexical profiling approach to comparing spoken and written corpora. In *Proceedings of the VIIth GSCP International Conference: Speech and Corpora*. Firenze University Press, 2012.

[41] C. Gerstenberger, N. Partanen, M. Rießler, and J. Wilbur. Instant Annotations – Applying NLP Methods to the Annotation of Spoken Language Documentation Corpora. In *Proceedings of the Third Workshop on Computational Linguistics for Uralic Languages*, pages 25–36, St. Petersburg, Russia. Association for Computational Linguistics, 2017. URL: https://www.aclweb.org/anthology/W17-0604.

[42] C. Gerstenberger, N. Partanen, M. Rießler, and J. Wilbur. Utilizing Language Technology in the Documentation of Endangered Uralic Languages. *Northern European Journal of Language Technology*, 4:29–47, 2016.

[43] Giellatekno - Saami Language Technology, UiT The Arctic University of Norway and The Divvun group at UiT The Arctic University of Norway. SIKOR North Saami free corpus, 2015. URL: http://hdl.handle.net/11509/100. Common Language Resources and Technology Infrastructure Norway (CLARINO) Bergen Repository.

[44] J. Giménez and L. Márquez. SVMTool: A general POS tagger generator based on Support Vector Machines. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC'04)*, 2004.

[45] J. Giménez and L. Márquez. SVMTool: A general POS tagger generator based on Support Vector Machines. Technical Manual v1.4, 2012. URL: https://www.cs.upc.edu/~nlp/SVMTool/SVMTool.v1.4.pdf (visited on 05/25/2020).

[46] S. T. Gries and A. L. Berez. Linguistic Annotation in/for Corpus Linguistics. In N. Ide and J. Pustejovsky, editors, *Handbook of Linguistic Annotation*, pages 379–409. Springer, 2017.

[47] T. Hagemeijer, M. Généreux, I. Hendrickx, A. Mendes, A. Tiny, and A. Zamora. The Gulf of Guinea Creole Corpora. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 523–529, Reykjavik, Iceland. European Language Resources Association (ELRA), May 2014. URL: http://www.lrec-conf.org/proceedings/lrec2014/pdf/438_Paper.pdf.

[48] S. Hattori. *Ainugo hōgen jiten* [Dictionary of Ainu dialects]. Iwanami Shoten, Tōkyō, 1964.

[49] J. He, Z. Zhang, T. Berg-Kirkpatrick, and G. Neubig. Cross-Lingual Syntactic Transfer through Unsupervised Adaptation of Invertible Projections. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3211–3223, Florence, Italy. Association for Computational Linguistics, 2019. DOI: 10.18653/v1/P19-1311.

[50] Hokkaidō Government, Environment and Lifestyle Section. Heisei nijū-kunen Hokkaidō Ainu seikatsu jittai chōsa hōkokusho [Report of the Survey on the Hokkaidō Ainu actual living conditions in 2017], 2017. URL: http://www.pref.hokkaido.lg.jp/ks/ass/H29_ainu_living_conditions_survey_.pdf (visited on 04/08/2020).

[51] Hokkaidō Utari Kyōkai [Hokkaidō Ainu Association]. *Akor Itak* [Our language]. Hokkaidō Utari Kyōkai, Sapporo, 1994.

[52] W. Huang, X. Cheng, K. Chen, T. Wang, and W. Chu. Toward Fast and Accurate Neural Chinese Word Segmentation with Multi-Criteria Learning. *CoRR*, abs/1903.04190, 2019.

[53] Information Resources Center, Research Institute for Languages and Cultures of Asia and Africa, Tokyo University of Foreign Studies. AA-ken Ainu-go shiryō kōkai purojekuto [Ainu Language Material Release Project], n.d. URL: http://ainugo.aa-ken.jp/ (visited on 07/21/2018).

[54] K. Izutsu. Ainu Languages: Correct, Good, and Less So. In M. Ishihara, E. Hoshino, and Y. Fujita, editors, *Self-determinable Development of Small Islands*, pages 269–285. Springer Singapore, Singapore, 2016. ISBN: 978-981-10-0132-1. DOI: 10.1007/978-981-10-0132-1_15. URL: https://doi.org/10.1007/978-981-10-0132-1_15.

[55] K. Jinbō and S. Kanazawa. *Ainugo kaiwa jiten* [Ainu conversational dictionary]. Kinkōdō Shoseki, Tōkyō, 1898.

[56] T. Katayama. *Ainu shin-yōshū o yomitoku* [Studying the *Ainu shin-yōshū*]. Katayama Institute of Linguistic and Cultural Research, 2003.

[57] D. Katō, H. Echizen'ya, K. Araki, Y. Momouchi, and K. Tochinai. Automatic Construction of the Bilingual Words Dictionary for Ainu-to-Japanese Using Recursive Chain-link-type Learning. In *Proceedings of the 1st Forum on Information Technology*, 2002.

[58] S. Kawaguchi and Y. Ma. Corrective Feedback, Negotiation of Meaning and Grammar Development: Learner-Learner and Learner-Native Speaker Interaction in ESL. *Open Journal of Modern Linguistics*, 2:57–70, 2012. DOI: 10.4236/ojml.2012.22008.

[59] S. Kayano. *Kayano Shigeru no Ainu shinwa shūsei* [A collection of Ainu myths by Shigeru Kayano] (vols. 1-10). Heibonsha, Tōkyō, 1998.

[60] S. Kayano. *Kayano Shigeru no Ainugo jiten* [Shigeru Kayano's Ainu dictionary]. Sanseidō, Tōkyō, 1996.

[61] G. Kazeminejad, A. Cowell, and M. Hulden. Creating lexical resources for polysynthetic languages—the case of Arapaho. In *Proceedings of the 2nd Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 10–18, Honolulu. Association for Computational Linguistics, Mar. 2017.

[62] A. Kilgarriff and G. Grefenstette. Introduction to the Special Issue on the Web as Corpus. *Computational Linguistics*, 29(3):333–347, Sept. 2003. ISSN: 0891-2017. DOI: 10.1162/089120103322711569. URL: https://doi.org/10.1162/089120103322711569.

[63] K. Kindaichi. *Ainu Jojishi, Yūkara no Kenkyū* [Studies of yukar, the Ainu epics]. Tōyō Bunko, Tōkyō, 1931.

[64] K. Kindaichi and M. Kannari. *Ainu Jojishi Yūkara-shū* [Collection of yukar, the Ainu epics], vols. 1-8. Sanseidō, Tōkyō, 1959/1968.

[65] H. Kirikae. *Ainu shin-yōshū jiten: tekisuto, bumpō kaisetsu tsuki* [Lexicon to Yukie Chiri's Ainu Shin-yōshū with text and grammatical notes]. Daigaku Shorin, Tōkyō, 2003.

[66] H. Kirikae. Ainu ni yoru Ainugo hyōki [Transcription of the Ainu language by Ainu people]. *Koku-bungaku kaishaku to kanshō* [Interpretation and appreciation of Japanese literature], 62(1):99–107, 1997.

[67] R. Kneser and H. Ney. Improved backing-off for M-gram language modeling. In *ICASSP*, pages 181–184. IEEE Computer Society, 1995. ISBN: 0-7803-2431-5. URL: http://dblp.uni-trier.de/db/conf/icassp/icassp1995.html#KneserN95.

[68] P. Koehn and H. Hoang. Factored Translation Models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 868–876, Prague, Czech Republic. Association for Computational Linguistics, June 2007. URL: https://www.aclweb.org/anthology/D07-1091.

[69] H. Kucera and W. N. Francis. *Computational analysis of present-day American English*. Brown University Press, Providence, RI, 1967.

[70] T. Kudo, K. Yamamoto, and Y. Matsumoto. Applying Conditional Random Fields to Japanese Morphological Analysis. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 230–237, Barcelona, Spain. Association for Computational Linguistics, 2004. URL: https://www.aclweb.org/anthology/W04-3230.

[71] Lewis, M.; Simons, G.; Fennig, C. (Eds.) Ethnologue: Languages of the World, Nineteenth edition, Dallas, Texas, 2016. URL: http://www.ethnologue.com.

[72] M. H. Long. The role of the linguistic environment in second language acquisition. In W. C. Ritchie and T. K. Bhatia, editors, *Handbook of second language acquisition*, pages 413–468. Academic Press, New York, 1996.

[73] J. Ma, K. Ganchev, and D. Weiss. State-of-the-art Chinese Word Segmentation with Bi-LSTMs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4902–4908, 2018.

[74] J. Ma, V. Henrich, and E. Hinrichs. Letter Sequence Labeling for Compound Splitting. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 76–81, Berlin, Germany. Association for Computational Linguistics, 2016.

[75] J. C. Maher. Akor Itak – Our Language, Your Language: Ainu in Japan. In J. A. Fishman, editor, *Can Threatened Languages be Saved?*, pages 323–349. Multilingual Matters, 2001. ISBN: 978-1-85359-706-0.

[76] A. Majewicz, editor. *The Collected Works of Bronisław Piłsudski.* Vols. 1–3. Mouton de Gruyter, 1998–2004.

[77] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, 2014.

[78] J. D. Martin, H. Johnson, B. Farley, and A. Maclachlan. Aligning and Using an English-Inuktitut Parallel Corpus. In *ParallelTexts@NAACL-HLT*, 2003.

[79] K. Martin. Aynu itak : on the road to Ainu language revitalization. *Media and Communication Studies*, 60:57–93, 2011. URL: https://eprints.lib.hokudai.ac.jp/dspace/bitstream/2115/47031/1/MSC60_005.pdf.

[80] K. Matsuura, S. Ueno, M. Mimura, S. Sakai, and T. Kawahara. Speech Corpus of Ainu Folklore and End-to-end Speech Recognition for Ainu Language. *ArXiv*, abs/2002.06675, 2020.

[81] A. Michaud, O. Adams, T. Cohn, G. Neubig, and S. Guillaume. Integrating Automatic Transcription into the Language Documentation Workflow: Experiments with Na Data and the Persephone Toolkit. *Language Documentation & Conservation*, 12:393–429, 2018.

[82] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean. Efficient Estimation of Word Representations in Vector Space. *CoRR*, abs/1301.3781, 2013.

[83] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information*

Processing Systems - Volume 2, NIPS'13, pages 3111–3119, Lake Tahoe, Nevada. Curran Associates Inc., 2013.

[84] T. Mikolov, W.-t. Yih, and G. Zweig. Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia. Association for Computational Linguistics, June 2013. URL: https://www.aclweb.org/anthology/N13-1090.

[85] Y. Momouchi. Incremental direct translation of noun phrases of the Ainu language to Japanese. In *IPSJ SIG Technical Report*, volume 162, pages 79–86, 2004.

[86] Y. Momouchi, Y. Azumi, and Y. Kadoya. Research note: Construction and utilization of electronic data for "Ainu shin-yōsyū". *Bulletin of the Faculty of Engineering at Hokkai-Gakuen University*, 35:159–171, 2008.

[87] Y. Momouchi and R. Kobayashi. Dictionaries and Analysis Tools for the Componential Analysis of Ainu Place Name. *Engineering Research: The Bulletin of Graduate School of Engineering at Hokkai-Gakuen University*, 10:39–49, 2010.

[88] W. Monroe, S. Green, and C. D. Manning. Word Segmentation of Informal Arabic with Domain Adaptation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 206–211, Baltimore, Maryland. Association for Computational Linguistics, 2014.

[89] H. Morita, D. Kawahara, and S. Kurohashi. Morphological Analysis for Unsegmented Languages using Recurrent Neural Network Language Model. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2292–2297, Lisbon, Portugal. Association for Computational Linguistics, 2015.

[90] K. Murasaki. *Karafuto Ainugo nyūmon kaiwa / First Step for the Sakhalin Ainu Language*. Ryokugeisha, Kushiro, 2009.

[91] M. Nagata. A self-organizing Japanese word segmenter using heuristic word identification and re-estimation. In *Fifth Workshop on Very Large Corpora*, 1997. URL: https://www.aclweb.org/anthology/W97-0120.

[92] H. Nakagawa. *Ainugo Chitose Hōgen Jiten* [Dictionary of the Chitose dialect of Ainu]. Sōfūkan, Tōkyō, 1995.

[93] H. Nakagawa. Ainu-jin ni yoru Ainugo hyōki e no torikumi [Efforts to transcribe the Ainu language by Ainu people]. In A. Shiohara and S. Kodama, editors, *Hyōki no shūkan no nai gengo no hyōki = Writing unwritten languages*, pages 1–44. Tōkyō gaikoku-go daigaku, Ajia/Afurika gengo bunka kenkyūjo [The Research Institute for Languages, Cultures of Asia, and Africa, Tokyo University of Foreign Studies], Tōkyō, 2006.

[94] H. Nakagawa. *Nyū ekusupuresu Ainugo*. Hakusuisha, Tōkyō, 2013.

[95] H. Nakagawa, A. Bugaeva, M. Kobayashi, and K. Kimura. Glossed Audio Corpus of Ainu Folklore, National Institute for Japanese Language and Linguistics, 2016. URL: http://ainucorpus.ninjal.ac.jp (visited on 12/10/2017).

[96] National Institute for Japanese Language and Linguistics. A Topical Dictionary of Conversational Ainu, 2015. URL: http://ainutopic.ninjal.ac.jp (visited on 08/25/2017).

[97] G. Neubig, Y. Nakata, and S. Mori. Pointwise Prediction for Robust, Adaptable Japanese Morphological Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 529–533, Portland, Oregon, USA. Association for Computational Linguistics, 2011. URL: https://www.aclweb.org/anthology/P11-2093.

[98] Nibutani Ainu Culture Museum. Ainu Language & Ainu Oral Literature, Nibutani Ainu Culture Museum, N.d. URL: http://www.town.biratori.hokkaido.jp/biratori/nibutani/culture/language/ (visited on 12/11/2017).

[99] J. Niehues and E. Cho. Exploiting Linguistic Resources for Neural Machine Translation Using Multi-task Learning. In *Proceedings of the Second Conference on Machine Translation*, pages 80–89, Copenhagen, Denmark. Association for Computational Linguistics, Sept. 2017. DOI: 10.18653/v1/W17-4708. URL: https://www.aclweb.org/anthology/W17-4708.

[100] P. Norvig. Natural language corpus data. In T. Segaran and J. Hammerbacher, editors, *Beautiful data*, pages 219–242. O'Reilly Media, Sebastopol, CA, 2009.

[101] K. Nowakowski, M. Ptaszynski, and F. Masui. A proposal for a unified corpus of the Ainu language. In *IPSJ SIG Technical Reports*, volume 2018-NL-237, pages 1–6, 2018.

[102] K. Nowakowski, M. Ptaszynski, and F. Masui. Improving Tokenization, Transcription Normalization and Part-of-speech Tagging of Ainu Language through Merging Multiple Dictionaries. In *Proceedings of the 8th Language & Technology Conference (LTC'17)*, pages 317–321, 2017.

[103] K. Nowakowski, M. Ptaszynski, and F. Masui. MiNgMatch - A Fast N-gram Model for Word Segmentation of the Ainu Language. *Information*, 10:317, 2019. DOI: 10.3390/info10100317.

[104] K. Nowakowski, M. Ptaszynski, and F. Masui. Word n-gram based tokenization for the Ainu language. In *Proceedings of International Workshop on Modern Science and Technology (IWMST 2018)*, pages 58–69, 2018.

[105] K. Nowakowski, M. Ptaszynski, F. Masui, and Y. Momouchi. Applying Support Vector Machines to POS tagging of the Ainu Language. In *Proceedings of the Workshop on Computational Methods for Endangered Languages*, volume 2, pages 17–23, 2019.

[106] K. Nowakowski, M. Ptaszynski, F. Masui, and Y. Momouchi. Framework for Ainu Language Processing Research. In *Language Acquisition and Understanding Research Group (LAU) Technical Reports*, pages 9–20, Sapporo, Japan, Feb. 2020.

[107] K. Nowakowski, M. Ptaszynski, F. Masui, and Y. Momouchi. Improving Basic Natural Language Processing Tools for the Ainu Language. *Information*, 10:329, 2019. DOI: https://doi.org/10.3390/info10110329.

[108] F. J. Och and H. Ney. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51, 2003.

[109] D. D. Palmer. A Trainable Rule-Based Algorithm for Word Segmentation. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 321–328, Madrid, Spain. Association for Computational Linguistics, 1997. URL: https://www.aclweb.org/anthology/P97-1041.

[110] C. P. Papageorgiou. Japanese Word Segmentation by Hidden Markov Model. In *Proceedings of the Workshop on Human Language Technology*, HLT '94, pages 283–288, Plainsboro, NJ. Association for Computational Linguistics, 1994. ISBN: 1-55860-357-3.

[111] W. Pei, T. Ge, and B. Chang. Max-Margin Tensor Neural Network for Chinese Word Segmentation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 293–303, Baltimore, Maryland. Association for Computational Linguistics, 2014.

[112] F. Peng, F. Feng, and A. McCallum. Chinese Segmentation and New Word Detection Using Conditional Random Fields. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04, Geneva, Switzerland. Association for Computational Linguistics, 2004.

[113] B. Peterson. Project Okikirmui. The complete Ainu legends of Chiri Yukie, in English, 2013. URL: http://www.okikirmui.com/ (visited on 09/17/2017).

[114] S. Petrov, D. Das, and R. McDonald. A Universal Part-of-Speech Tagset. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2089–2096, Istanbul, Turkey. European Language Resources Association (ELRA), May 2012. URL: http://www.lrec-conf.org/proceedings/lrec2012/pdf/274_Paper.pdf.

[115] B. Plank, S. Klerke, and Z. Agic. The Best of Both Worlds: Lexical Resources To Improve Low-Resource Part-of-Speech Tagging. *ArXiv*, abs/1811.08757, 2018.

[116] B. Plank, A. Søgaard, and Y. Goldberg. Multilingual Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Models and Auxiliary Loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 412–418, Berlin, Germany. Association for Computational Linguistics, Aug. 2016. DOI: 10.18653/v1/P16-2067. URL: https://www.aclweb.org/anthology/P16-2067.

[117] M. Ptaszynski, Y. Ito, K. Nowakowski, H. Honma, Y. Nakajima, and F. Masui. Combining Multiple Dictionaries to Improve Tokenization of Ainu Language. In *Proceedings of The 31st Annual Conference of the Japanese Society for Artificial Intelligence*, 2017.

[118] M. Ptaszynski, F. Masui, and Y. Momouchi. A Toolkit for Analysis of Ainu Language. In *Demo Session of 20th International Conference on Language Processing and Intelligent Information Systems (LP&IIS 2013)*, 2013.

[119] M. Ptaszynski and Y. Momouchi. Part-of-Speech Tagger for Ainu Language Based on Higher Order Hidden Markov Model. *Expert Systems With Applications*, 39:11576–11582, 14, 2012.

[120] M. Ptaszynski and Y. Momouchi. POST-AL: Part-of-Speech Tagger for Ainu Language. In *Proceedings of The 18th Annual Meeting of The Association for Natural Language Processing (NLP-2012)*, pages 763–766, 2012.

[121] M. Ptaszynski, K. Mukaichi, and Y. Momouchi. NLP for Endangered Languages: Morphology Analysis, Translation Support and Shallow Parsing of Ainu Language. In *Proceedings of The 19th Annual Meeting of The Association for Natural Language Processing*, pages 418–421, 2013.

[122] M. Ptaszynski, K. Nowakowski, Y. Momouchi, and F. Masui. Comparing Multiple Dictionaries to Improve Part-of-Speech Tagging of Ainu Language. In *Proceedings of The 22nd Annual Meeting of The Association for Natural Language Processing*, pages 973–976, 2016.

[123] M. Ptaszynski, P. Dybala, R. Rzepka, K. Araki, and Y. Momouchi. YACIS: A Five-Billion-Word Corpus of Japanese Blogs Fully Annotated with Syntactic and Affective Information. In *Proceedings of the AISB/IACAP 2012 Symposium: Linguistic And Cognitive Approaches To Dialogue Agents* (LaCATODA 2012), 2012.

[124] I. Radliński. Słownik narzecza Ainów zamieszkujących wyspę Szumszu w łańcuchu Kurylskim przy Kamczatce, ze zbiórow Prof. B. Dybowskiego [A dictionary of the dialect of Ainu inhabiting the Shumshu Island in the Kurile Archipelago near Kamchatka, collected by prof. B.Dybowski]. In *Rozprawy Akademii Umiejętności, Wydział Filologiczny, serya II, tom I*. Kraków, 1892.

[125] N. Randall. A Survey of Robot-Assisted Language Learning (RALL). In *ACM Transactions on Human-Robot Interaction*, volume 9 of number 1, Dec. 2019. DOI: https://doi.org/10.1145/3345506.

[126] K. Refsing. *The Ainu language. The morphology and syntax of the Shizunai dialect*. Aarhus University Press, Aarhus, 1986.

[127] T. Ruokolainen, O. Kohonen, S. Virpioja, and M. Kurimo. Supervised Morphological Segmentation in a Low-Resource Learning Setting using Conditional Random Fields. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 29–37, Sofia, Bulgaria. Association for Computational Linguistics, Aug. 2013.

[128] Sapporo Gakuin Daigaku, Jinbungaku-bu [Sapporo Gakuin University, Faculty of Humanities]. *Ainu bunka ni manabu* [Learning from Ainu culture]. Sapporo Gakuin Daigaku Seikatsu Kyōdō Kumiai, Ebetsu, 1990.

[129] M. Sassano. Deterministic Word Segmentation Using Maximum Matching with Fully Lexicalized Rules. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, pages 79–83, Gothenburg, Sweden. Association for Computational Linguistics, 2014. URL: https://www.aclweb.org/anthology/E14-4016.

[130] M. Sato, H. Ouchi, and Y. Tsuboi. Addressee and Response Selection for Multilingual Conversation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3631–3644, Santa Fe, New Mexico, USA. Association for Computational Linguistics, Aug. 2018. URL: https://www.aclweb.org/anthology/C18-1308.

[131] T. Satō. *Ainugo bunpō no kiso* [Basics of Ainu grammar]. Daigaku Shorin, Tōkyō, 2008.

[132] T. Satō. Ainugo Chitose hōgen ni okeru meishi-hōgō: sono shurui to kanren sho-kisoku [Noun incorporation in the Chitose dialect of Ainu: types and related rules]. *Hokkaidō-ritsu Ainu Minzoku Bunka Kenkyū Sentā Kenkyū Kiyō / Bulletin of the Hokkaidō Ainu Culture Research Center*, 18:1–32, 2012.

[133] T. Satō. Ainugo kobunken kenkyū to jōhō shori [Information processing in the study of historical Ainu language documents]. In S. Katō and T. Satō, editors, *Jōhō kagaku to gengo kenkyū* [Information science in linguistic research], pages 147–172. Gendai Tosho, 2016.

[134] H. Senuma and A. Aizawa. Toward Universal Dependencies for Ainu. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 133–139, 2017.

[135] H. Senuma and A. Aizawa. Universal Dependencies for Ainu. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, pages 2354–2358, 2018.

[136] B. Settles. Active Learning Literature Survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[137] Y. Shao, C. Hardmeier, and J. Nivre. Universal Word Segmentation: Implementation and Interpretation. *CoRR*, abs/1807.02974, 2018.

[138] Y. Shao, C. Hardmeier, J. Tiedemann, and J. Nivre. Character-based Joint Segmentation and POS Tagging for Chinese using Bidirectional RNN-CRF. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 173–183, Taipei, Taiwan. Asian Federation of Natural Language Processing, 2017.

[139] M. Shibatani. *The languages of Japan*. Cambridge University Press, London, 1990.

[140] G. Simons and C. F. (Eds.) Ethnologue: Languages of the World, Twenty-first edition. Dallas, Texas, 2018. URL: http://www.ethnologue.com.

[141] K. Sunasawa. *Ku sukup oruspe* [My life story]. Miyama Shobō, Sapporo, 1983.

[142] O. Täckström, D. Das, S. Petrov, R. McDonald, and J. Nivre. Token and Type Constraints for Cross-Lingual Part-of-Speech Tagging. *Transactions of the Association for Computational Linguistics*, 1:1–12, 2013. DOI: 10.1162/tacl_a_00205. URL: https://www.aclweb.org/anthology/Q13-1001.

[143] S. Tamura. *Ainugo jiten: Saru hōgen. The Ainu-Japanese Dictionary: Saru dialect*. Sōfūkan, Tōkyō, 1996.

[144] S. Tamura. Ainugo [The Ainu language]. In T. Kamei, R. Konō, and E. Chino, editors, *Gengo-gaku daijiten serekushon: Nihon rettō no gengo* [Selection from the Encyclopedia of Linguistics: Languages of the Japanese archipelago]. Sanseidō, Tōkyō, 1997.

[145] The Ainu Museum. Ainu-go Ākaibu [Ainu Language Archive], 2017–2020. URL: http://ainugo.ainu-museum.or.jp/ (visited on 08/15/2018).

[146] The Foundation for Ainu Culture. *Chūkyū Ainugo – Saru* [Intermediate Ainu course – Saru dialect]. The Foundation for Ainu Culture, Sapporo, 2014.

[147] D. Tuggener. Evaluating Neural Sequence Models for Splitting (Swiss) German Compounds. In M. Cieliebak, D. Tuggener, and F. Benites, editors, *Proceedings of the 3rd Swiss Text Analytics Conference - SwissText 2018, Winterthur, Switzerland, June 12-13, 2018*, volume 2226 of *CEUR Workshop Proceedings*, pages 42–49. CEUR-WS.org, 2018. URL: http://ceur-ws.org/Vol-2226.

[148] K. Uehara and C. Abe. *Ezo hōgen moshiogusa* [Ezo dialect dictionary]. 1804.

[149] UNESCO ad Hoc Expert Group on Endangered Languages. Language Vitality and Endangerment. 2003. URL: http://www.unesco.org/culture/ich/doc/src/00120-EN.pdf (visited on 12/23/2017).

[150] M. van Gompel and M. Reynaert. FoLiA: A practical XML format for linguistic annotation – a descriptive and comparative study. *Computational Linguistics in the Netherlands Journal*, 3:63–81, Dec. 2013. URL: https://clinjournal.org/clinj/article/view/26.

[151] K. Wang, C. Thrasher, and B.-J. P. Hsu. Web Scale NLP: A Case Study on URL Word Breaking. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 357–366, Hyderabad, India. ACM, 2011. ISBN: 978-1-4503-0632-4. URL: http://doi.acm.org/10.1145/1963405.1963457.

[152] P. Wang, Y. Qian, F. K. Soong, L. He, and Z. Hai. Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Recurrent Neural Network. *ArXiv*, abs/1510.06168, 2015.

[153] Z. Wu and G. Tseng. Chinese Text Segmentation for Text Retrieval: Achievements and Problems. *Journal of the American Society for Information Science*, 44(9):532–542, 1993. ISSN: 0002-8231.

[154] N. Xue. Chinese Word Segmentation as Character Tagging. In *International Journal of Computational Linguistics & Chinese Language Processing, Volume 8, Number 1, February 2003: Special Issue on Word Formation and Chinese Language Processing*, pages 29–48, 2003.

[155] J. Yang, Y. Zhang, and F. Dong. Neural Word Segmentation with Rich Pretraining. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 839–849, 2017.

[156] J. Yang, Y. Zhang, and S. Liang. Subword Encoding in Lattice LSTM for Chinese Word Segmentation. *CoRR*, abs/1810.12594, 2018.

[157] D. Yarowsky and G. Ngai. Inducing Multilingual POS Taggers and NP Bracketers via Robust Projection across Aligned Corpora. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies*, NAACL '01, pages 1–8, Pittsburgh, Pennsylvania. Association for Computational Linguistics, 2001. DOI: 10.3115/1073336.1073362. URL: https://doi.org/10.3115/1073336.1073362.

[158] Y. Zhang, D. Gaddy, R. Barzilay, and T. Jaakkola. Ten Pairs to Tag – Multilingual POS Tagging via Coarse Mapping between Embeddings. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1307–1317, San Diego, California. Association for Computational Linguistics, June 2016. DOI: 10.18653/v1/N16-1156. URL: https://www.aclweb.org/anthology/N16-1156.

[159] H. Zhao, C.-N. Huang, M. Li, and B.-L. Lu. Effective Tag Set Selection in Chinese Word Segmentation via Conditional Random Field Modeling. In *Proceedings of the 20th Pacific Asia Conference on Language, Information and Computation*, pages 87–94, Huazhong Normal University, Wuhan, China. Tsinghua University Press, 2006.

[160] X. Zheng, H. Chen, and T. Xu. Deep Learning for Chinese Word Segmentation and POS Tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 647–657, Seattle, Washington, USA. Association for Computational Linguistics, 2013.

Research Achievements

First Author Publications

1. K. Nowakowski, M. Ptaszynski, and F. Masui. Towards Better Text Processing Tools for the Ainu Language. In *Lecture Notes in Artificial Intelligence*. Springer Verlag, 2020. (in press).
2. K. Nowakowski, M. Ptaszynski, F. Masui, and Y. Momouchi. Framework for Ainu Language Processing Research. In *Language Acquisition and Understanding Research Group (LAU) Technical Reports*, pages 9–20, Sapporo, Japan, Feb. 2020.
3. K. Nowakowski, M. Ptaszynski, and F. Masui. MiNgMatch - A Fast N-gram Model for Word Segmentation of the Ainu Language. *Information*, 10:317, 2019. DOI: 10.3390/info10100317.
4. K. Nowakowski, M. Ptaszynski, F. Masui, and Y. Momouchi. Applying Support Vector Machines to POS tagging of the Ainu Language. In *Proceedings of the Workshop on Computational Methods for Endangered Languages*, volume 2, pages 17–23, 2019.
5. K. Nowakowski, M. Ptaszynski, F. Masui, and Y. Momouchi. Improving Basic Natural Language Processing Tools for the Ainu Language. *Information*, 10:329, 2019. DOI: https://doi.org/10.3390/info10110329.
6. K. Nowakowski, M. Ptaszynski, and F. Masui. A proposal for a unified corpus of the Ainu language. In *IPSJ SIG Technical Reports*, volume 2018-NL-237, pages 1–6, 2018.

7. K. Nowakowski, M. Ptaszynski, and F. Masui. Word n-gram based tokenization for the Ainu language. In *Proceedings of International Workshop on Modern Science and Technology (IWMST 2018)*, pages 58–69, 2018.
8. K. Nowakowski, M. Ptaszynski, and F. Masui. Improving Tokenization, Transcription Normalization and Part-of-speech Tagging of Ainu Language through Merging Multiple Dictionaries. In *Proceedings of the 8th Language & Technology Conference (LTC'17)*, pages 317–321, 2017.

Co-authored Publications

1. J. Nieuważny, F. Masui, M. Ptaszynski, K. Araki, R. Rzepka, and K. Nowakowski. Emotional and moral impressions associated with buddhist religious terms in japanese blogs-preliminary analysis. In A. V. Samsonovich, editor, *Biologically Inspired Cognitive Architectures 2019*, pages 387–392, Cham. Springer International Publishing, 2020. ISBN: 978-3-030-25719-4.
2. J. Nieuważny, F. Masui, M. Ptaszynski, R. Rzepka, and K. Nowakowski. How religion and morality correlate in age of society 5.0: statistical analysis of emotional and moral associations with buddhist religious terms appearing on japanese blogs. *Cognitive Systems Research*, 59:329–344, 2020. ISSN: 1389-0417. DOI: https://doi.org/10.1016/j.cogsys.2019.09.026. URL: http://www.sciencedirect.com/science/article/pii/S138904171930498X.
3. J. Nieuważny, F. Masui, M. Ptaszynski, R. Rzepka, and K. Nowakowski. Preliminary Statistical Analysis of Emotional and Moral Impressions Associated with Buddhist Religious Terms. In *Proceedings of International Workshop on Modern Science and Technology (IWMST 2018)*, pages 51–57, 2018.
4. M. Ptaszynski, Y. Ito, K. Nowakowski, H. Honma, Y. Nakajima, and F. Masui. Combining Multiple Dictionaries to Improve Tokenization of Ainu Language. In *Proceedings of The 31st Annual Conference of the Japanese Society for Artificial Intelligence*, 2017.

5. M. Ptaszynski, K. Nowakowski, Y. Momouchi, and F. Masui. Comparing Multiple Dictionaries to Improve Part-of-Speech Tagging of Ainu Language. In *Proceedings of The 22nd Annual Meeting of The Association for Natural Language Processing*, pages 973–976, 2016.

Awards

1. Outstanding Research Award[1] at the 237th Meeting of the Special Interest Group on Natural Language Processing of the Information Processing Society of Japan (September 25-26, 2018) for:

   K. Nowakowski, M. Ptaszynski, and F. Masui. A proposal for a unified corpus of the Ainu language. In *IPSJ SIG Technical Reports*, volume 2018-NL-237, pages 1–6, 2018.

2. 2019 IPSJ Yamashita SIG Research Award[2] for:

   K. Nowakowski, M. Ptaszynski, and F. Masui. A proposal for a unified corpus of the Ainu language. In *IPSJ SIG Technical Reports*, volume 2018-NL-237, pages 1–6, 2018.

[1] https://nl-ipsj.or.jp/award/
[2] https://www.ipsj.or.jp/award/yamashita2019.html