ChaKi.NET liteの開発

国立国語研究所論集 (NINJAL Research Papers) 25: 75–88 (2023)
ISSN 2186-1358

ChaKi.NET lite の開発

伊藤　薫[a]　　森田敏生[b]

[a] 九州大学／国立国語研究所 共同研究員
[b] 総和技研

要旨

現代において，コーパスは言語研究に欠かせない資源となっている。言語学の分野では検索・閲覧・集計インターフェイスを備えたコーパスの利用が多いが，情報学等の分野で作成されたコーパスには必ずしもインターフェイスが提供されるわけではない。類型論研究での活用が期待される Universal Dependencies（UD）ツリーバンクもそのようなコーパスの 1 つである。そこで本研究では，既存の高機能コーパスツールである ChaKi.NET を情報抽出用に特化し，新規ユーザにも利用しやすい軽量版である ChaKi.NET lite を開発した。ChaKi.NET は高機能であるがゆえに利用者にとっての学習コストが高かったが，ChaKi.NET lite では UD に合わせたインターフェイスを提供し，アノテーション機能を省くことで目的の機能を利用しやすくした。本稿では ChaKi.NET lite 開発の背景と機能について紹介する *。

キーワード：ChaKi.NET，Universal Dependencies（UD），コーパスツール

1. はじめに

現代において，コーパスは自然言語処理や言語学研究のために幅広く利用されている。言語学研究では，何らかの理論や仮説を検証するためにコーパスを用いたり，コーパス自体を言語についての仮説を構築する手段としたりする（McEnery and Hardie 2012: 5–6）。一方，自然言語処理では，コーパスは主に機械学習用の訓練・評価データとして用いられる。自然言語処理では与えられたタスクに応じて生テキストに様々なアノテーションが施されることがあるが，中には言語学研究との親和性が高いコーパスが存在する。自然言語処理用のコーパスは機械処理を前提としているため，人間がそのデータを検索したり閲覧したりする際に使いやすいツールが付属していることは稀である。そのため，このようなデータを活用するためには，容易に利用可能な情報抽出のためのツールが必要である。本稿では，既存のコーパスツールである ChaKi.NET のインターフェイスをライトユーザ向けに改良した ChaKi.NET lite 開発の経緯や同ツールの位置づけ，機能について述べる。

* 本稿の一部は国立国語研究所の共同研究プロジェクト「実証的な理論・対照言語学の推進」（プロジェクトリーダー：浅原正幸）の研究成果である。また，本研究は JSPS 科研費 19K13180 の助成を受けたものである。なお，本稿は Evidence-based Linguistics Workshop 2022 で発表した際の予稿「ChaKi.NET lite の開発：Universal Dependencies コーパスの利用を見据えた ChaKi.NET ユーザインターフェイスの改良」（伊藤・森田 2022）を加筆・修正したものである。本研究にあたり，共同研究プロジェクトを取りまとめてくださった浅原正幸先生，ツール構築に関して示唆的な意見をくださった木本幸憲氏，中馬隼人氏，田中悠介氏，ならびに Evidence-based Linguistics Workshop 2022 においてコメントをくださった方々に感謝申し上げたい。

1.1 Universal Dependencies に関わる動向

自然言語処理向けのコーパス群として，近年 Universal Dependencies（UD）が注目を集めている。UD は通言語的に統一された方法で依存構造や品詞などについてアノテーションが付与されたコーパス群を開発する国際プロジェクトである（Nivre et al. 2016）。特徴としては依存構造が内容語主辞で付与されていること，Universal POS（UPOS）と呼ばれる品詞体系や Universal Dependency Relations と呼ばれる依存関係タグが定義され，コーパスに付与されていることなどがあげられる。また，人手アノテーションのしやすさやパージング精度の高さなど，言語学的妥当性以外も考慮して設計されている（Nivre et al. 2016）。

UD は 2017 年の初回（UDW 2017）（de Marneffe et al. 2017）から執筆時点までに 5 回のワークショップが行われており，直近のワークショップ（UDW 2021）（de Lhoneux and Tsarfaty 2021）では 15 件の発表が行われるなど，研究開発が盛んなプロジェクトである。また，公開されてから既に 3 年以上が経過した日本語 UD に関する論文（浅原ほか 2019）が本稿執筆時点（2022 年 11 月）においても J-STAGE における『自然言語処理』誌の月間アクセス数トップ（2022 年 10 月）であることからも，国内の自然言語処理分野でも UD への関心が高いことが窺える。

言語学研究における UD ツリーバンクの利点は，主にデータの多様さにある。2022 年 11 月時点での最新バージョン（Ver. 2.11）では 138 言語, 243 ツリーバンクを収録している。これにはチュクチ語やワルビリ語などの少数言語，アッカド語やコプト語，古代中国語（いわゆる漢文）等の古代語，ヒンディー語／英語やトルコ語／ドイツ語などのコードスイッチング，学習者コーパス，手話（スウェーデン手話）などが含まれ，個々のデータ量は少ない場合もあるが時空間的，様態的に非常に幅広い言語が収録されている。加えて，多くのデータがフリーで利用できることも多言語データへの気軽なアクセスを可能にしている。

近年，自然言語処理分野だけでなく，言語類型論分野でも UD ツリーバンクの活用は進んでいる。従来，World Atlas of Language Structures（WALS）（Dryer and Haspelmath eds. 2013）ではカテゴリーや順序に基づく分類が行われており，タイプ基盤（type-based）の研究が行われていた。これに対し，token-based typology（Levshina 2019），corpus-based typology（Schnell and Schiborr 2022; Levshina 2022）など様々な呼称があるが，コーパスを利用して実際の言語使用における特定の言語単位や構造の観察によって一般化や分類を試みる動きが出てきている。これらの研究においても UD が用いられており，通言語的に共通した品詞体系と依存関係をアノテーションするという特徴が活かされている。このような動向から，UD は自然言語処理だけでなく，言語類型論をはじめとした言語学の諸分野でも有用であると期待できる。

1.2 既存の ChaKi.NET と UD ツリーバンク検索ツール

ChaKi.NET（Matsumoto et al. 2006）は自然言語コーパスの構築，検索，閲覧，アノテーションが可能な高機能コーパスツールである。その機能には文字列検索，係り受け検索といった基本的な機能に加え，ワードリスト検索，MI スコアや N グラムの取得といったコロケーションに関わる情報の取得機能も含まれる。係り受け検索クエリは GUI を介して記述できるため，全て文

字で記述するタイプのクエリに比べ学習コストが低く済むという利点がある。また，UD ツリーバンクのデータ形式である CoNLL-U 形式のファイルも ChaKi.NET 上の機能を使い SQLite の .db 形式に変換することで読み込み可能であるため，UD ツリーバンクを読み込み言語学研究に用いることも可能である。

しかし，その高機能さと歴史の長さゆえに ChaKi.NET は新規ユーザにとって利用しづらいツールとなっている側面もある。機能面では上記のようなコーパス利用者側のための機能のほか，アノテーション機能をはじめとしたコーパス作成者のための機能も含まれている。また，現在の ChaKi.NET は 2009 年にリリースされたバージョンから連続性を保ちつつ機能向上されているが，画面レイアウトやボタン配置は開発当初からあまり変更されていないため，近年の一般的なインターフェイスに慣れた新規ユーザには馴染みにくい部分もある。したがって，より容易に UD ツリーバンクを使用できるようにするには，ChaKi.NET の機能を活かしつつインターフェイスを見直す必要がある。

執筆時点で UD 公式サイトで紹介されているサードパーティのツールのうち，検索を主な機能としたツールは TüNDRA（Martens 2013）と UDeasy（Villa 2022）があげられる。このうち，TüNDRA はブラウザ上で利用可能なツールで，UD ツリーバンクもブラウザ上で直接選択できる。しかし，TIGERSearch query と呼ばれるクエリを入力する必要があり，複雑な係り受けを検索するにはクエリの記述に習熟する必要がある。UDeasy は CoNLL-U ツリーバンクを検索できるオフラインツールで，GUI を介してクエリ入力できる。しかし，兄弟関係（sibling）の検索が可能という利点はあるものの，クエリ入力インターフェイスはドラッグにより係り受け関係を指定する ChaKi.NET ほど直感的であるとはいえず，検索結果のツリーも CUI で表示されるため統語木を観察することには不向きである。

このような背景から，本研究では ChaKi.NET をもとに UD の処理に適した軽量版を開発することで，より容易に情報抽出・閲覧が可能なツールの開発を目指す。

2. ChaKi.NET lite の開発方針

2.1 UD の特徴とツリーバンクのデータ構造

UD は言語学，特に少数言語研究や類型論分野で有用なコーパスだが，自然言語処理分野で開発されたコーパスであるため，利用者がプログラミングスキルを持つことを想定している。このため自分でスクリプトを書くことができれば必要な情報を抽出できるが，そうでない場合は何らかのツールを利用する必要がある。本節では，一般的な利用者が UD を利用する際に障壁になりそうな点について紹介する。

1.1 で述べたように UD は同じ枠組みでデータが記述されているため横断的な検索が容易である。つまり，品詞を例にあげると UD では UPOS と呼ばれる統一された品詞体系でアノテーションされているため，一般名詞と接置詞（前置詞・後置詞）の係り受けなどを検索したい場合は全ての言語で NOUN タグと ADP タグの係り受けを検索すればよく，個別の言語に合わせてクエリを記述する煩雑さが軽減される。また，UD ツリーバンクは既存の別形式の係り受けのコーパ

スや句構造文法のコーパスを変換して作成されることが多い。UD では語に対し XPOS と呼ばれる個別言語に特化した品詞情報を付与することができるが，変換元の高品質な品詞・形態論情報を XPOS として保持しているツリーバンクも多く，個別言語に特化した検索も可能である。

UD ツリーバンクの各コーパスは CoNLL-U（拡張子は .conllu）というフォーマットで書かれている（図 1）が，これはある程度人間が直接読むことのできる形式ではあるものの，可視化ツールを介さずに目視で多くの依存構造木や品詞情報を見ながら分析することには適していない。例えば，図 1 の情報をツリー表示した図 2 の方が，人間がツリーを読むのに適していることは明らかである。また，テキスト形式（.txt）のデータも同梱されているため多くの例文を読むこと，テキストエディタを用いて例文を検索することは可能だが，.conllu ファイルとは異なりコーパステキスト本文の情報しか含まれないため，品詞情報や依存構造に関する情報は利用できない。

```
# newdoc id = train-s2118
# sent_id = train-s2118
# text = 右にある[表示]をクリックすると一覧表示される。
1 右 右 NOUN　名詞-普通名詞-一般_ 3 obl _ BunsetuBILabel=B|BunsetuPosition-
Type=SEM_HEAD|LUWBILabel=B|LUWPOS=名詞-普通名詞-一般|SpaceAfter=No|UnidicInfo=,右,
右,右,ミギ,,,ミギ,ミギ,右
2 にに ADP 助詞-格助詞_ 1 case _ BunsetuBILabel=I|BunsetuPositionType=SYN_HEAD|LUW-
BILabel=B|LUWPOS=助詞-格助詞|SpaceAfter=No|UnidicInfo=,に,に,に,ニ,,,ニ,ニ,に
3 ある 有る VERB　動詞-非自立可能-五段-ラ行_ 5 acl _ BunsetuBILabel=B|BunsetuPosition-
Type=SEM_HEAD|LUWBILabel=B|LUWPOS=動詞-一般-五段-ラ行|PrevUDLemma=ある|SpaceAf-
ter=No|UnidicInfo=,有る,ある,ある,アル,,,アル,アル,有る
4 [ [ PUNCT 補助記号-括弧開 _ 5 punct _ BunsetuBILabel=B|BunsetuPosition-
Type=CONT|LUWBILabel=B|LUWPOS=補助記号-括弧開|SpaceAfter=No|UnidicInfo=,［,［,［,,,,,,
［
5 表示 表示 NOUN　名詞-普通名詞-サ変可能 _ 8 obj _ BunsetuBILabel=I|BunsetuPosition-
Type=SEM_HEAD|LUWBILabel=B|LUWPOS=名詞-普通名詞-一般|SpaceAfter=No|UnidicInfo=,表示,
表示,表示,ヒョージ,,,ヒョウジ,ヒョウジ,表示
6 ] ] PUNCT 補助記号-括弧閉 _ 5 punct _ BunsetuBILabel=I|BunsetuPosition-
Type=CONT|LUWBILabel=B|LUWPOS=補助記号-括弧閉|SpaceAfter=No|UnidicInfo=,］,］,］,,,,,,］
7 ををADP 助詞-格助詞_ 5 case _ BunsetuBILabel=I|BunsetuPositionType=SYN_HEAD|LUW-
BILabel=B|LUWPOS=助詞-格助詞|SpaceAfter=No|UnidicInfo=,を,を,を,オ,,,ヲ,ヲ,を
```

図 1　CoNLL-U 形式のデータ（UD_Japanese-GSD）

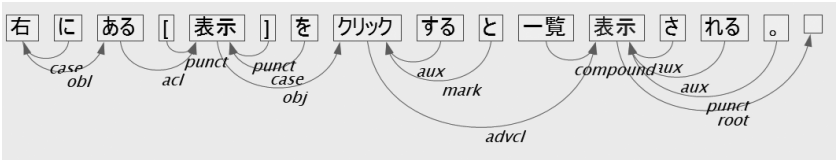

図 2　ツリー表示されたデータ（UD_Japanese-GSD）

次に，UD ツリーバンクのフォルダ構造について示す。1.1 で述べたように最新の UD ツリーバンク（ver. 2.11）には 138 言語，243 ツリーバンクが収録されている。収録されている言語の数とツリーバンクの数からわかるように，1 言語に対し複数のツリーバンクが利用できる場合がある。例えば，図 3 中列に並んでいるフォルダは個々のツリーバンクに対応しており，ツリーバンク名内の “UD_” に続く部分が個別言語名を示す。ここでは，アバザ語，アフリカーンス語のツリーバンクは 1 つずつだが，日本語は少なくとも 6 ツリーバンクが利用できる [1] ことがわかる。同じ言語に属するツリーバンクでもアノテーション方法が異なる場合があるため，1 つのツリーバンクに範囲を絞り検索することにも利点はある。しかし，細かなアノテーションの差異よりも 1 言語からできるだけ多くの用例を得たい場合など，複数のツリーバンクを横断的に検索したい場合も考えられる。

また，収録語数の多い言語では機械学習に用いやすいようデータが最大 dev，test，train の 3 つに分割されている。例えば，図 3 右列ではアフリカーンス語のツリーバンク “Af_AfriBooms” が dev，test，train という 3 つの .conllu ファイルに分割されている。このため，機械学習ではなく単に品詞情報や依存構造情報を持ったアノテーション済みコーパスとして用いたい場合は，3 つのファイル全てから情報を検索することが望まれるが，そのためには複数のファイルを横断し

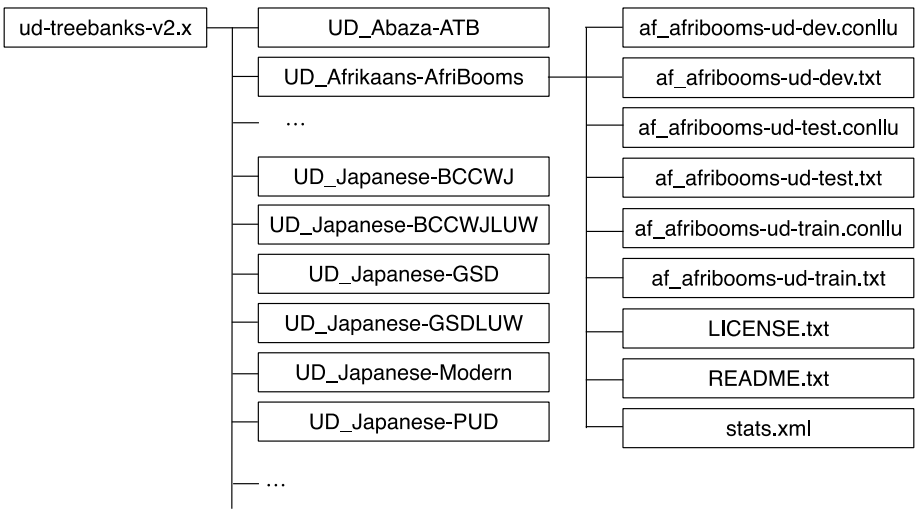

図 3　UD ツリーバンクのフォルダ構造

[1] 実際には 6 つ以上の日本語ツリーバンクが含まれるが，ここでは紙幅の都合上 6 つのみを表示している。

て検索しなければならないという点で煩雑である。

2.2 ChaKi.NET lite 開発の目的と方針

2.1 で述べたように，UD は言語学や関連分野の研究にとって有用なデータであるものの，何らかのインターフェイスを利用するか，自身でスクリプトを作成しなければ必要な情報を抽出することができない。本研究では想定する中心的ユーザ像を Web 版の COCA や中納言など，インターフェイス付きコーパスの利用に慣れた言語学研究者とするため，スクリプトよりも言語研究に適した何らかのインターフェイスを提供する方が望ましいと考える。

本研究で開発する ChaKi.NET lite のベースとなる既存の ChaKi.NET は，習熟した利用者にとっては高機能で使いやすいものの，高機能ゆえに操作が複雑であるため，新規ユーザにとって学習コストが高い状況となっている。こうした状況をふまえ，本研究では CoNLL-U の読み込みやタグ，依存構造木表示，依存構造情報の検索など，UD コーパスの利用に適した基本的機能を備える ChaKi.NET を改良することで，言語学や関連分野の研究を促進するツールの開発を目指す。また，ChaKi.NET をベースとすることで，ChaKi.NET が持つ多くの機能のうち言語学研究に適した機能を継承することも可能である。これらをふまえ，本研究では ChaKi.NET lite 開発の方針を以下のように定めた。

・インターフェイスの改良により直感的に利用可能であること
・ChaKi.NET のデータベースと互換性があること
・言語学研究に適した機能を持つこと
・コーパスを利用して言語学研究を行うユーザを主な対象とすること
・UD ツリーバンクの操作に適していること

なお，UD ツリーバンクの利用は ChaKi.NET lite 開発の契機ではあるが，用途は UD ツリーバンクの使用に限らない。ChaKi.NET lite は所定の SQLite 形式のコーパスであれば読み取り可能であるため，手元のコーパスも SQLite 形式に変換することで利用可能である。また，既存 ChaKi.NET をアノテーション機能を含んだ高機能バージョン，ChaKi.NET lite をコンコーダンサ特化の新規ユーザ向け軽量版と位置づけ，両者を併存させる。

3. ChaKi.NET lite における新要素

3.1 機能

3.1.1 ツリーバンク一括読み込み

ChaKi.NET では .conllu ファイルを読み込む際，アプリケーション上の機能を用いて SQLite ファイルに変換する操作が必要であった。この操作は中納言を介した『現代日本語書き言葉均衡コーパス』の利用，あるいは，ウェブ上のインターフェイスが提供されている BNC，COCA を始めとしたコーパスを中心に利用している研究者にとっては使用上の障壁になる。特に，UD では 2.1 節で説明したデータ構造が ChaKi.NET でツリーバンクを読み込む上での障壁になる。例えば，

データが機械学習向けに分割されているため dev，train，test という最大 3 つのファイルを統合して .SQLite 形式に変換する操作が必要である。既存の ChaKi.NET では，複数の .conllu ファイルを入れたフォルダを指定し，ChaKi.NET 上のツールを用いてそのフォルダ内のファイルを結合し単一の SQLite 形式ファイルに変換できる。しかし，この操作は 1 ツリーバンクにつき 1 度実行しなければいけないため，複数のツリーバンクを使用したい場合はツリーバンクの数だけ同様の操作が必要となる。これは多数のツリーバンクから例文を検索したい利用者にとって大きな負担となる。

そこで，ChaKi.NET lite では UD ツリーバンクデータから事前に変換･提供される SQLite ファイルの格納されたフォルダを画面上（次頁の図 4 左列）にドラッグ & ドロップして一括読み込みする機能を追加した。これにより，複雑な前処理なしで大規模な通言語コーパスの利用を可能にした。なお，現バージョンの ChaKi.NET lite では提供された変換済みファイルを読み込む方式をとっているが，この方式では UD ツリーバンク本体のバージョンアップへの対応等，本体側の変化に対応することが難しい。そのため，ユーザ側でツリーバンクを SQLite 形式に変換する方法を何らかの形で提供することが望ましいと考えている。変換処理にはある程度時間がかかるため，ドラッグ & ドロップ時にこの変換を自動実行することは好ましくない。したがって，現時点では変換用のバッチファイルの開発・配布が望ましいと考えている。

3.1.2 ツリーバンク選択

言語類型論や対照研究などの言語横断的な研究をする場合は，複数言語のツリーバンクの中から検索対象を選択し，時には切り替える必要がある。このような操作の必要性を考慮し，ChaKi.NET から検索対象コーパスを選択する機能を大幅に改良した。ChaKi.NET lite におけるツリーバンク選択画面は，図 4 左列（「コーパス読込・選択」と示した部分）に当たる。元の ChaKi.NET では検索クエリを入力する SearchConditions パネルにタブとして組み込まれていたが，検索対象コーパスの可視性や切り替えやすさを重視し，独立したエリアとして常時表示させる仕様とした。ここに示すように，各ツリーバンクは言語ごとにまとめられており，言語名をクリックすることで折り畳み・展開が可能である。検索対象コーパスは各コーパス名のクリックにより切り替え可能であり，検索対象となっている場合はチェックボックスにチェックが入り，コーパス名が赤字で強調表示される。

3.2 インターフェイスの改善

3.2.1 画面レイアウト

ChaKi.NET lite では複数のコーパスを検索対象とし，必要に応じて検索範囲を切り替えながらクエリを入力して例文を検索するという流れを想定した画面レイアウトとした（図 4）。既存の ChaKi.NET ではユーザが自分の好みや用途に合わせて各パネルを配置できたが，ChaKi.NET lite では簡便に使えることを目標としているため，アノテーションを利用した検索が可能なコンコーダンサとしての機能に特化して予め各エリアを配置した。

まず，操作を大域的操作と局所的操作に分け，大域的操作では 3.1 で述べたコーパス読み込み・選択を実行できる。図 4 右列に配置した各エリアでは局所的操作が可能である。上段で検索クエリを入力した後，検索（Search ボタン）を実行すると中段に検索結果が表示される。中段エリアでは検索における中心語が赤色，中心語以外として入力した語が青色で強調表示される。用例をダブルクリックすると下段に統語構造木を表示できる。このように，大域的操作として検索対象コーパスを指定した後，右列上段から下段へ向かうにしたがって徐々に詳細な情報が表示できるよう配置している。

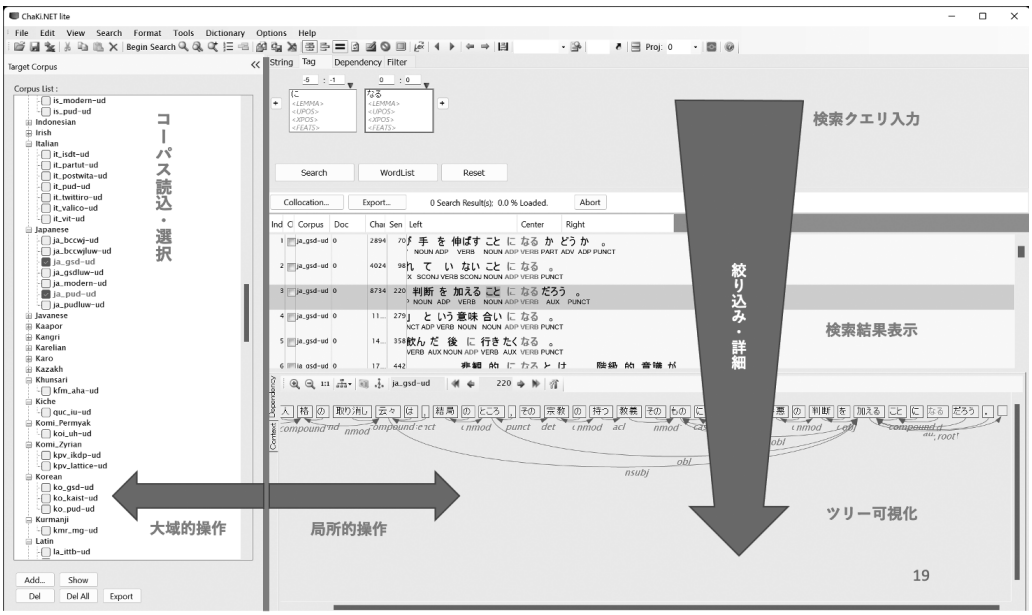

図 4　ChaKi.NET lite のインターフェイス

3.2.2 ボタン配置変更による操作性の改善

3.2.1 で紹介した全体的な配置以外にも，ChaKi.NET lite ではコンコーダンサとしての操作がスムーズになるようボタン配置を変更した。既存の ChaKi.NET では本節で述べる機能が主に従来ウィンドウ最上部のツールバーに設置されていたが，検索窓の横に検索ボタンが配置されている Google 検索など，最近のツールに多く見られるインターフェイスに慣れているユーザには目的の処理を行うボタンが発見しづらい配置だと思われる。そこで，ChaKi.NET lite では上段から下段に行くに従い詳細になる全体のエリア配置方針に従い，ボタンも各エリアと配置場所を関連させ，下段に行くほど詳細に関する操作となるようデザインした。以下では，配置を変更した機能について配置変更の目的と機能の詳細を説明する。

最上段の検索エリアには，検索（Search）機能，WordList 機能，Reset 機能を配置した。検索機能では条件にマッチした結果を KWIC 形式で返す一般的な検索を実行できる。WordList 機能は条件にマッチした結果を異なり語ごとに集計し表として出力する。Reset 機能を使用すると入

力クエリを全消去し初期状態に戻すことができる。これらのうち，WordList 機能について詳しく説明すると，レンマ，UPOS など語のプロパティの一部分をマッチング条件として指定し，語の異なりごとの頻度を表形式で出力する機能である。例えば，be というレンマ，ADJ という品詞が連続することを条件にして検索した場合，*m* 件の “be able”，*n* 件の “is good”，…といった条件に合致する検索結果の頻度の一覧表を返す。一覧表では各要素の表層形（FORM），レンマ（LEMMA），UPOS，XPOS，Feats（features），各コーパスでの頻度，全コーパスでの頻度，全体における出現比率に関する情報が出力され，画面上で並べ替えができる。ChaKi.NET では SearchConditions パネルで上記のような検索条件を設定し，Command パネルと呼ばれる別のパネルで WordList ボタンを押すことで当該機能を実行することができた。しかし，検索条件を設定するパネルと WordList 機能実行が別パネルに配置されているため，新規ユーザに存在を認識しづらい機能となっている。そこで, コンコーダンサ機能を中心としている ChaKi.NET lite では，条件設定後すぐに視界に入る検索条件入力箇所の直下に WordList 実行ボタンを配置した。

また，検索機能や WordList を実行した後に用いる検索結果表示エリアの上には，統計情報を取得するための Collocation ボタン，結果をエクスポートするための Export ボタン，検索ヒット数や進捗状況を示す表示，検索が終了しない場合に手動で終了させる Abort ボタンを配置した。Collocation 機能は基本的に KWIC（Search 機能の実行結果）に対して利用でき，Collocation ボタンを押すと表示される別ウィンドウで条件を指定し実行することで粗頻度（Raw Frequency），相互情報量（MI）スコア，*n*-gram，パターン抽出（FSM, Frequent Sequence Mining）の情報を取得することができる。Collocation 機能により粗頻度の情報を取得すると，Search の条件として設定した中心語（タグ検索や係り受け検索で赤い枠で囲まれた語）を基準に，そこからの相対位置ごとに語の頻度を確認することができる。MI スコアは中心語とその周りに現れる語との共起の強さに関する指標を算出する機能である。実行すると機能名にもなっている MI スコアのほか，MI3 スコア，Dice 係数，log-log スコア，z スコアといった共起の強さに関する複数の指標値が表示される。*n*-gram 機能を使用すると，中心語から見て *n* 語の範囲で連続する表層形ごとの頻度を取得できる。なお，*n*-gram を取得する際は中心語から見て左右どちらに向けた情報を取得するかを指定できる。FSM は PrefixSpan（Pei et al. 2001）と呼ばれるアルゴリズムに基づいて文集合に対する頻出語列（パターン）をマイニングする機能である。他の機能とは異なり，FSM では KWIC の結果を用いない。これらの Collocation 機能はすべて ChaKi.NET から引き継いでいる。エクスポート機能を使用すると KWIC データ，WordList データ，Collocation 各機能の出力データを様々な形式のファイルとして保存できる。KWIC データは Microsoft Excel ファイルもしくは CSV ファイル，WordList, Collocation の出力は Microsoft Excel ファイル，CSV ファイルに加え，R の data.frame としてもエクスポート可能である。WordList, Collocation をエクスポートする場合，CSV または R を選択するとクリップボードにコピーされる。

なお，本節で紹介した WordList 機能の内容については浅原・森田（2015）でより詳しい情報を参照できる。

3.2.3 各パネルの改良

ChaKi.NET lite では，アノテーションツールとしての機能も持つ ChaKi.NET から主にアノテーション機能を削減し，コンコーダンサとしての役割に特化している。また，UD ツリーバンクの使用を想定した改良も施している。本節ではこれらの目的に合わせて変更した各エリア（ChaKi.NET ではパネル（panel）と呼ばれる）の改良点について述べる。

まず，ChaKi.NET における SearchConditions パネルに相当する検索エリアについては，エリア内のタブ構成と検索対象フィールドを改良した。ChaKi.NET では図 5 左に示す通り，SearchConditions パネルでは Corpus，Filter，String，Tag，Dependency，Collocation，Add-in の 7 個のタブが配置されていたが，ChaKi.NET lite では String，Tag，Dependency，Filter の 4 つに簡略化した。SearchConditions パネルをコーパス選択エリアとして独立させたことで Corpus タブは不要となり，検索実行後に行う Collocation 処理は操作の流れを重視して結果表示エリア上部に配置したため Collocation タブも不要となった。また，Add-in 機能は高機能版である ChaKi.NET の利用者向けであるため削除した。

また，検索フィールドも UD 向けに簡素化した。既存の ChaKi.NET のフィールド（図 5 左）では，タグ検索，依存関係検索では表層形，読みなど様々なアノテーション情報を指定して検索できるが，UD では用いられていないフィールドも多く含まれている。この仕様は新規ユーザにとって目的の機能を見つける際の妨げになるため，CoNLL-U で規定されているフィールドのみを表示するよう変更した（図 5 右）。

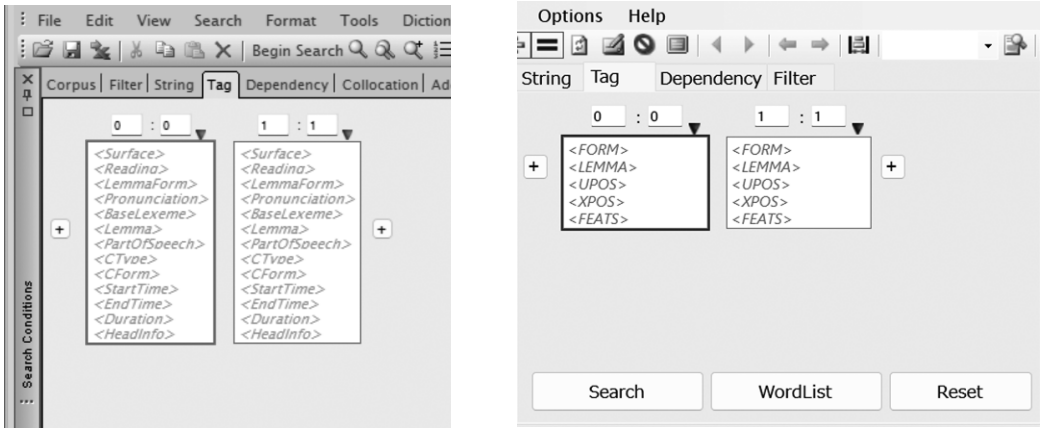

図 5　ChaKi.NET の検索条件パネル（左）と ChaKi.NET lite の検索条件エリア（右）

次に，ツリー表示エリアではアノテーション機能をなくし，コンコーダンサとしての機能への特化により可視性を向上させた。実際の画面を図 6 に示す。まず，アノテーション機能を削除し，コンコーダンサとして不要な情報を表示させないように変更した。これにより，図 6 上段で表示されている語の枠線や，新たな語を追加するための「+」ボタンを削除して，下段のようにより

多くの語を表示できるようにした。また，従来，語をクリックすると別パネルで表示されていた語彙情報は，マウスオーバーにより表示するよう変更した。下段では，スクリーンショットに写り込んでいないが「に」の部分に重なっており，マウスオーバーされている間は語彙情報（Lemma, upos など）が表示される。

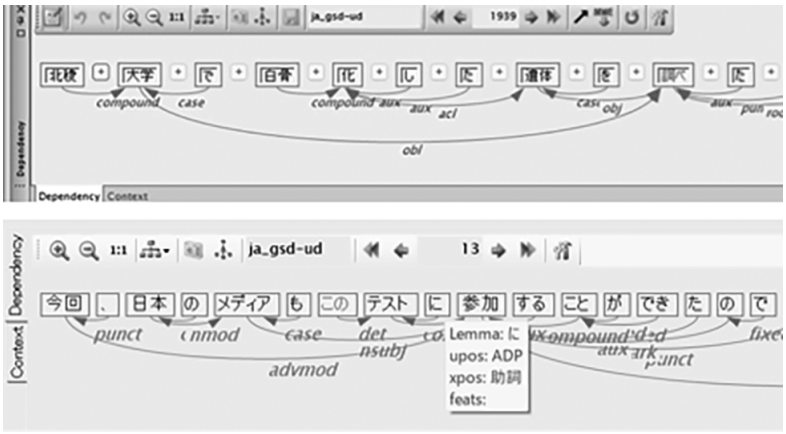

図 6　ChaKi.NET のツリー表示（上段）と ChaKi.NET lite のツリー表示（下段）

3.3 UD ツリーバンク使用上の注意点

3.3.1 依存関係矢印の方向

UD は Tesnière（1959）に端を発する依存文法に基づいている。依存文法では，主要部が従属部を従えるという関係で語と語の統語関係を捉えるため，統語木で構造を表す際は主要部から従属部に向かう矢印で表示する。これは日本語で一般的に用いられる，係り元から係り先へ向かう矢印とは逆向きとなる。既存の ChaKi.NET では一般的な日本語の係り受け表示に従い，デフォルトでは UD の概念でいう従属部から主要部の向きで矢印が描画される。これに対し，ChaKi.NET lite では UD で一般的な表示法に従い，主要部から従属部への矢印をデフォルトとした。既存の ChaKi.NET，本研究で開発した ChaKi.NET lite ともに矢印の向きの設定は利用者の需要に合わせ反転できるが，UD の矢印の向きが日本語の係り受けと逆であることには注意されたい。

3.3.2 著作権とコーパス本文

UD ツリーバンクには，テキスト本文は含まれずアノテーション情報のみが提供される場合もある。例えば, UD の公式サイトからダウンロードできるデータでは, 日本語 UD で最大のツリーバンクである UD_Japanese-BCCWJ の本文は著作権上の都合により削除されており，UD で付加されたアノテーション情報のみが配布されている。当該コーパスの完全な情報を利用するには，BCCWJ のデータを入手し，UD ツリーバンクに同梱されているスクリプトを実行して本文の情報を復元するか，BCCWJ 公式サイトで所有者向けに公開されている UD ツリーバンクをダウン

ロードする必要がある。また，当該コーパスを ChaKi.NET や ChaKi.NET lite で利用するには復元済みの .conllu ファイルを SQLite 形式に変換する必要がある。

他にも，UD_Hindi_English-HIENCS も同様に，公式サイトからダウンロードできるデータには本文が含まれない。当該コーパスは Twitter のデータを元に作成されており，復元には同梱のスクリプトを用いて Twitter のデータをクロールする必要がある。ChaKi.NET lite は簡便な UD ツリーバンク検索ツールの提供により言語学や関連分野への貢献を目標としているが，このような事情で使用するためにはコマンドラインを介した操作が要求されるツリーバンクも含まれる。また，オンラインで利用可能な UD ツリーバンクの検索プログラムも存在するが，オンラインでは著作権の問題により基本的にはオンラインサービスに組み込んでデータを提供できない。ChaKi.NET lite のようなオフラインツールであれ，コーパスファイルをアップロード可能なオンラインツールであれ，これらのコーパスを利用するには復元操作が必要である。

4. おわりに

4.1 結語と展望

本研究では，主に UD ツリーバンクを言語学や関連分野の研究に使用する際に障害となる要因を取り除くべく，ChaKi.NET の軽量版として ChaKi.NET lite を開発した。本論文では，その主な機能や改良点を解説した。

ChaKi.NET lite は UD ツリーバンクだけでなく，今後開発される他の CoNLL 形式や SQLite 形式のデータを利用する際にも利用できる。今後は UD ツリーバンクに限らず，様々な研究領域で開発される多種多様な言語資源が言語学や関連分野の研究に活用されることを期待する。

4.2 プログラムの配布

ChaKi.NET lite は GitHub 上にて Windows 上で動作するソースコード，ビルド済みバイナリおよびインストーラが公開されている。使用にはソースコードからのビルドを行う必要はなく，ビルド済みバイナリを展開するか，またはインストールを行うだけで利用できる。

参照文献

浅原正幸・金山博・宮尾祐介・田中貴秋・大村舞・村脇有吾・松本裕治（2019）「Universal Dependencies 日本語コーパス」『自然言語処理』26(1): 3–36.

浅原正幸・森田敏生（2015）「コーパスコンコーダンサ『ChaKi.NET』の「文書－部分構造行列」出力機能」『第 8 回コーパス日本語学ワークショップ予稿集』257–264. 東京：国立国語研究所.

伊藤薫・森田敏生（2022）「ChaKi.NET lite の開発：Universal Dependencies コーパスの利用を見据えた ChaKi.NET ユーザインターフェイスの改良」『Evidence-based Linguistics Workshop 発表論文集』1–5. 東京：国立国語研究所.

Dryer, Matthew S. & Martin Haspelmath (eds.) (2013) *The World Atlas of Language Structures Online (WALS)*. Leipzig: Max Planck Institute for Evolutionary Anthropology. https://wals.info (Accessed December 2022).

Levshina, Natalia (2019) Token-based typology and word order entropy: A study based on Universal Dependencies. *Linguistic Typology* 23: 533–572.

Levshina, Natalia (2022) Corpus-based typology: Applications, challenges and some solutions. *Linguistic Typology* 26: 129–160.

de Lhoneux, Miryam and Reut Tsarfaty (2021) *Proceedings of the Fifth Workshop on Universal Dependencies (UDW, SyntaxFest 2021)*. Association for Computational Linguistics, Sofia, Bulgaria.

de Marneffe, Marie-Catherine, Joakim Nivre and Sebastian Schuster (2017) *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*. Association for Computational Linguistics, Gothenburg, Sweden.

Martens, Scott (2013) TüNDRA: A web application for treebank search and visualization. *Proceedings of The Twelfth Workshop on Treebanks and Linguistic Theories (TLT12)*, 133–144, Sofia, Bulgaria.

Matsumoto, Yuji, Masayuki Asahara, Kiyota Hashimoto, Yukio Tono, Akira Ohtani and Toshio Morita (2006) An annotated corpus management tool: ChaKi. *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, 1418–1421, Genoa, Italy.

McEnery, Tony and Andrew Hardie (2012) *Corpus linguistics: Method, theory and practice*. Cambridge: Cambridge University Press.

Nivre, Joakim, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty and Daniel Zeman (2016) Universal dependencies v1: A multilingual treebank collection. *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 1659–1666, Portorož, Slovenia.

Pei, Jian, Jiawei Han, Behzad Mortazavi-Asi, Helen Pinto, Qiming Chen, Umeshwar Dayal and Mei-Chun Hsu (2001) PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth. *Proceedings of the 17th International Conference on Data Engineering*, 215–224.

Schnell, Stefan and Nils Norman Schiborr (2022) Crosslinguistic corpus studies in linguistic typology. *Annual Review of Linguistics* 8(1): 171–191.

Tesnière, Lucien (1959) *Eléments de syntaxe structurale*. Paris: Klincksieck.

Villa, Brigada Luca (2022) UDeasy: A tool for querying treebanks in CoNLL-U format. *Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-10)*, 16–19, Marseille, France.

関連 Web サイト

CLARIN-D. *TüNDRA*. https://weblicht.sfs.uni-tuebingen.de/Tundra/（2022 年 12 月 4 日確認）

Davies, Mark. *Corpus of Contemporary American English (COCA)*. https://www.english-corpora.org/coca/（2022 年 12 月 4 日確認）

Masayuki, Asahara, Toshio Morita, Masakazu Iwatate and Yuji Matsumoto.『ChaKi.NET』. https://ja.osdn.net/projects/chaki/（2022 年 11 月 24 日確認）

Universal Dependencies contributors. *Universal Dependencies*. https://universaldependencies.org/（2022 年 11 月 24 日確認）

Villa, Luca Brigada (2022) *UDeasy*. https://unipv-larl.github.io/udeasy/user_guide.html（2022 年 12 月 4 日確認）

伊藤薫・森田敏生『ChaKi.NET lite』. https://github.com/chakidev/chakinet-lite（2022 年 11 月 24 日確認）

国立国語研究所 コーパス検索アプリケーション『中納言』. https://chunagon.ninjal.ac.jp/（2022 年 11 月 24 日確認）

国立国語研究所『現代日本語書き言葉均衡コーパス（BCCWJ）』. https://clrd.ninjal.ac.jp/bccwj/（2022 年 11 月 24 日確認）

Developing ChaKi.NET Lite

ITO Kaoru[a]　　MORITA Toshio[b]

[a]Kyushu University / Project Collaborator, NINJAL
[b]Sowa Research

Abstract
Corpora are indispensable resources for contemporary linguistic research. While corpora used for linguistics research usually have an interface, those developed for informatics research tend to lack one. Universal Dependencies (UD) Treebank, which has proved useful for linguistic typology studies, also lacks an interface. In this study, we developed a lightweight corpus tool named ChaKi.NET lite for new users, specialized from the existing sophisticated ChaKi.NET for information extraction. While one takes a long time to learn to use ChaKi.NET, ChaKi.NET lite reduces the required learning time by omitting the annotation function and providing an interface tailored to UD. This paper introduces the background of the development of ChaKi.NET lite and the new functions that this corpus tool provides.

Keywords: ChaKi.NET, Universal Dependencies (UD), corpus tool