

UIMA を基盤とする相互運用性の向上と自動組み合わせ比較 —国際共同プロジェクト U-Compare—

狩野 芳伸[†] 辻井 潤一[†]

[†] 東京大学情報理工学系研究科コンピュータ科学専攻 〒113-0033 東京都文京区本郷 7-3-1

E-mail: [†] {kano, tsujii}@is.s.u-tokyo.ac.jp

あらまし 近年、利用可能な自然言語処理リソースは増加しているが、リソースの開発において相互運用性が考慮されることは少ない。一方で、自然言語処理のタスクは本質的にモジュールの組み合わせと考えられるものが多く、ツールの再利用・組み合わせ・比較・評価等を容易にする相互運用性の向上が必要とされている。本稿では、相互運用性向上のための汎用オープンフレームワークである UIMA を基盤として、ツールの自動組み合わせ比較を行うシステムと視覚的なユーザインターフェースの提供、および対応ツール・コーパスを収集公開する国際共同プロジェクト U-Compare について報告する。U-Compare.org ではシステムを Web 経由で自動配布しており、ワンクリックで起動できる。

キーワード UIMA, 自然言語処理ツール, 相互運用性, 組み合わせ比較, Web サービス

U-Compare: Interoperability and Combinatorial Comparison Based on UIMA

Yoshinobu KANO[†] and Jun'ichi TSUJII[†]

[†] Graduate School of Information Science and Technology, University of Tokyo 7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-0033 Japan

E-mail: [†] {kano, tsujii}@is.s.u-tokyo.ac.jp

Abstract Recently, the number of NLP resources, including tools and corpora, are increasing. However, such resources are developed independently in different groups without considering the interoperability. Because NLP tasks are composite in nature, the interoperability between resources is required to reuse, combine, compare and evaluate them. We have started an international joint project called U-Compare, which is built on top of UIMA, an open framework to provide the interoperability. U-Compare provides a system to perform combinatorial comparisons, graphical user interfaces, and ready-to-use U-Compare compatible resources. The U-Compare system can be launched by a single click, automatically distributed and deployed via the Internet.

Keyword UIMA, NLP tools, Interoperability, Combinatorial comparison, Web service

1. はじめに

構文解析器や品詞タガー、NER (Named Entity Recognizer)、タグ付きコーパス等、利用可能な自然言語処理リソースは増加している。一方で、自然言語処理リソースを組み合わせることでタスクを達成することは難しいことが多い。それは、入出力フォーマットの不一致のみならず、これまでのリソース開発では考慮されることの少なかった、より高度な相互運用性が必要とされるからである。

UIMA (Unstructured Information Management Architecture) は、そのような相互運用性向上のため IBM が開発した汎用フレームワーク [1] で、現在は仕様を OASIS UIMA 委員会 [2] で策定し、参照実装を Apache UIMA [3] として提供するオープンフレームワークにな

っている。

UIMA は柔軟性があり充実した機能をもつ優れたフレームワークであるが、UIMA 自体はあくまで汎用のフレームワークであることが目的で、UIMA に準拠したツールそのものや、UIMA においてデータの意味を表すための型を定義する type system は UIMA 標準としては提供されておらず、今後もされる予定がない。そのため、単にツールを UIMA 準拠にするだけでは十分な相互運用性が得られない。実際、UIMA 準拠のツールキットは、Carnegie Mellon 大学 [4]、Colorado 大学 [5]、Jena 大学 [6]、東京大学 [7] などから提供されているが、別々のグループで開発されたツールは通常 type system に互換性がないため、そのままでは組み合わせで動作させることができない。

さらに、そのような互換性が得られたとしても、別

個のグループが開発したツールは、たとえば同じ品詞タガーであっても大なり小なり異なる特徴をもつため、目的とするタスクにより最適なツール、最適な組み合わせは異なり、ユーザが最適なものを見つけ出すのは困難な作業である。

また、UIMA 関連のユーティリティーは、現状では開発者向けが主であり、一般ユーザにとって使いやすさとは言い難い。相互運用性向上の大きな利点の一つは、ユーザにとって本質的でない作業を軽減することにより質量の両面でタスクを改善することであり、ユーザビリティを欠くことはこの利点を大きく損なう。

これらの問題点を克服するため、我々は、UIMA 準拠のツールに適用可能な自動組み合わせ比較システムを開発し、異なる type system 間をつなぐ sharable type system の設計を提案してきた[8, 9]。さらに、我々東京大学と、NaCTeM (英国 National Centre for Text Mining) および CCP (Center for Computational Pharmacology, Colorado 大学 Health Science Center) との国際共同プロジェクトとして U-Compare[10]を立ち上げた。U-Compare では、広範囲の概念をカバーする sharable type system を提供し、その type system に互換な UIMA 準拠リソース群を収集公開すると同時に、ユーザビリティを向上させる様々なツール群を配布しており、ワンクリックでダウンロードから起動までを実行できる。

本稿では、執筆時の Apache UIMA に基づいて 2 章で UIMA について概説し、3 章で UIMA における前述の様々な問題点を克服した U-Compare プロジェクトについて報告する。

2. UIMA

UIMA では、すべてのデータは CAS (Common Analysis Structure) と呼ばれるデータ構造に格納される。CAS は生データとそれに付加された素性構造からなり、自然言語処理の場合、これらは生テキストと解析済み情報に相当する。UIMA では、生データは不変の固定データとし、そこにさまざまなツールが素性構造として解析情報を追加していくという考え方をとる。そのため、付加された素性構造は通常、生データのどの部分に付加されたのかを表すオフセット begin/end 値を持つ。これは一般に stand-off annotation 形式と呼ばれる。

CASの素性構造は汎用のグラフ構造であるが、すべての素性は型付けされ、開発者が定義したtype systemのいずれかのtypeをもつ。UIMAのtype systemはEMF (Eclipse Modeling Framework) Ecore互換である¹。

¹ 現在の Apache UIMA では、Ecore で許されている多重継承が実装されておらず、木構造である。

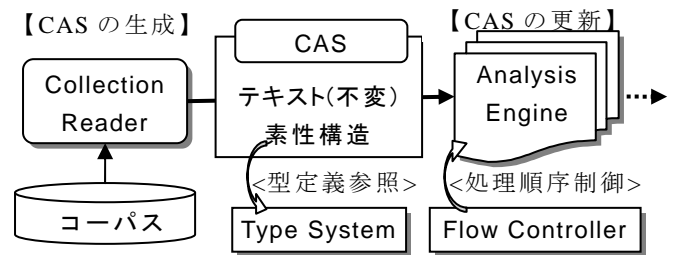


図 1. UIMA における処理と機構の概念図

CAS は抽象的なデータ構造であり、ファイルとして保存する場合は通常 XMI 形式で表現され、pure java の参照実装である Apache UIMA では実行中は java オブジェクトとして格納される。

CAS に対して何らかの処理を行う UIMA コンポーネントには、大きく分けて Collection Reader と Analysis Engine の二種類がある。Collection Reader はコーパス等のデータを読み込み、CAS を生成する。タグ付きコーパス等の場合、生成時に素性構造を付加することもある。Analysis Engine は生成済みの CAS を受け取り、何らかの処理を行い素性構造を追加するコンポーネントであり、品詞タガーなどのツールに相当する。Collection Reader および Analysis Engine は、どの type の素性を受け取り、どの type の素性を出力しうるかを Capability と呼ばれる属性として持つ。

Analysis Engine には、実際の処理を行う primitive と、入れ子可能な子コンポーネント群を保持する aggregate の 2 タイプがある。aggregate の場合、子コンポーネント群はパイプライン的に順に処理されることが多いが、UIMA でこの処理順序を制御する Flow Controller は完全にプログラマブルであり、基本的にはほとんどあらゆる動作が実現できる。

UIMA で扱う対象はテキストに限らず、画像・音声などさまざまなものを扱えるよう汎用に設計されている。同時に、対訳、音声と書き言葉の対応付けなど、対象の多面的な表現が自然にできるように、Sofa (Subject OF Analysis) という仕組みが用意されている。

UIMAの任意のコンポーネントは、SOAPによるWebサービスとして展開できる²。ユーザはWebサービスであることを意識することなく、ローカルサービスと混在したワークフローを実行できる。

3. U-Compare

2 章で紹介したように UIMA は優れたフレームワークであるが、不足点も数多くある。本章では、UIMA に準拠しつつ、さらに相互運用性とユーザビリティを

² Apache UIMA では、次期バージョンとして JMS (Java Message Service) を用いた非同期システムの開発が行われている。

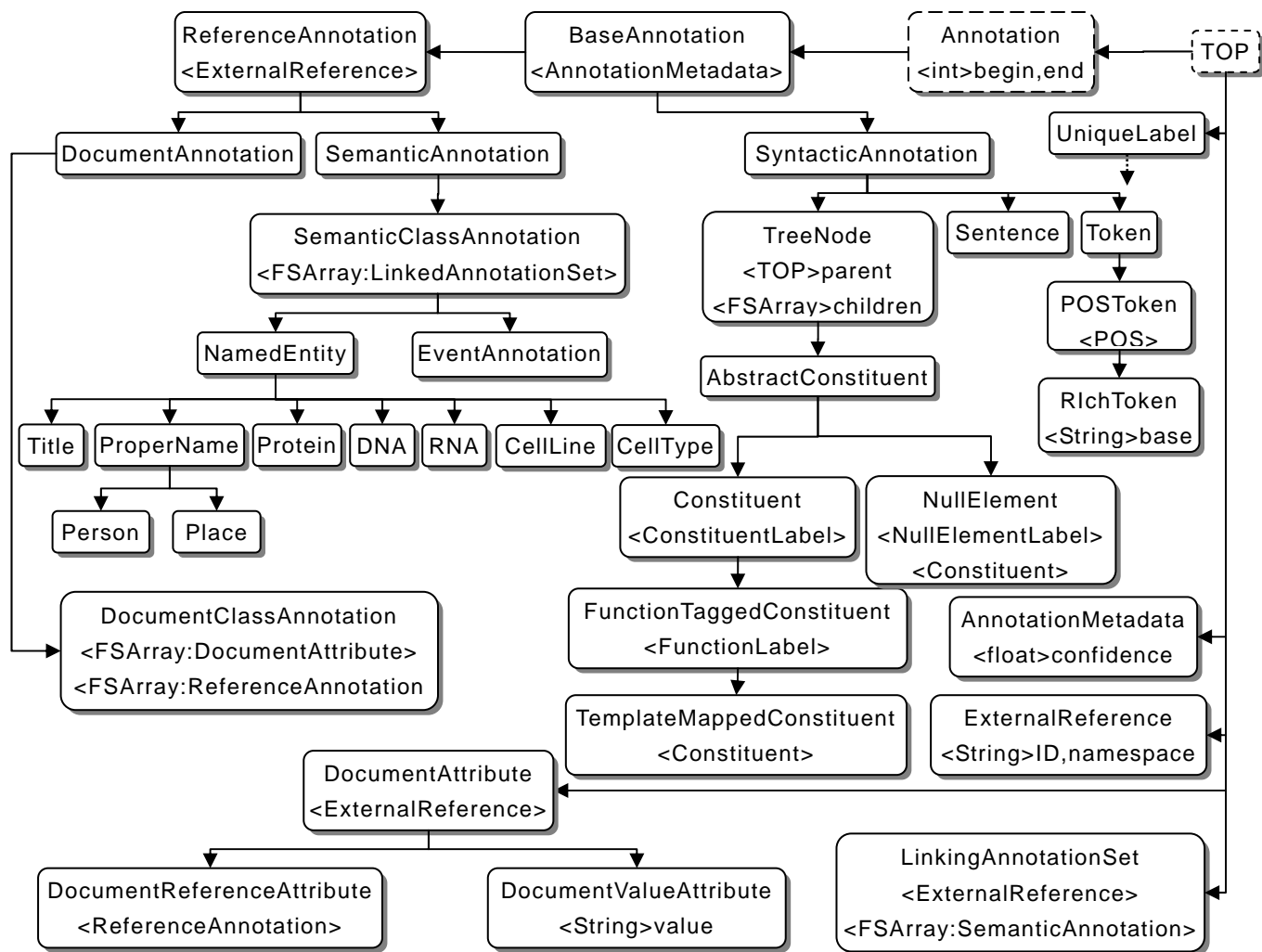


図 2. U-Compare type system 概念図 (UniqueLabel 以下のラベルを表す type は省略した)

向上させた U-Compare プロジェクトについて、sharable type system (3.1 節)、自動組み合わせ比較(3.2 節)、ユーザビリティとユーザインターフェース(3.3 節)、対応コンポーネントと Web サービス(3.4 節)、の順に説明する。

3.1. Sharable Type System

UIMA 準拠のツール同士であっても、type system が共通でなければそのまま組み合わせ実行することはできない。すべての開発者が同一の type system を使用していれば問題にならないが、「文」「単語」といった頻用される概念の統一的な定義すら難しいように、単一の type system は現実的な解決策とはいえない。

一方で、異なる type system が現れるたびに type system 変換器を作成するのでは、せっかくの相互運用性が損なわれる。であれば、異なる type system 間の架け橋となる sharable type system を用いるのが、自然な解決策であろう³。

type system 間の変換とは、似ているが同一ではない概念間の変換を行うことに相当し、多かれ少なかれ情報の損失は免れ得ないが、損失を最低限に抑えられる設計の sharable type system が望ましい。

また、type を使用する本質的な意義は一意性の保証にある。たとえば、type を用いずにあらゆる情報を単一の文字列素性として表すことは可能ではあるが、一意性が損なわれ、相互運用性を著しく低下させる。

しかし、細かい概念をどこまで type として定義する必要があるかは、特定のタスクに依存する上、定まった解答のある問題ではない。

U-Compare では、これらの点を踏まえて、統語・意味・文章構造など広範囲の概念をカバーしつつ極力一意性を保証できるように type system を設計した(図 2)。

変換ではなく多重継承を用いることができれば、変換のオーバーヘッドを避けられる上、多数の概念体系を同時に利用できる利点がある。

³ sharable type system を使う場合、type system 間の

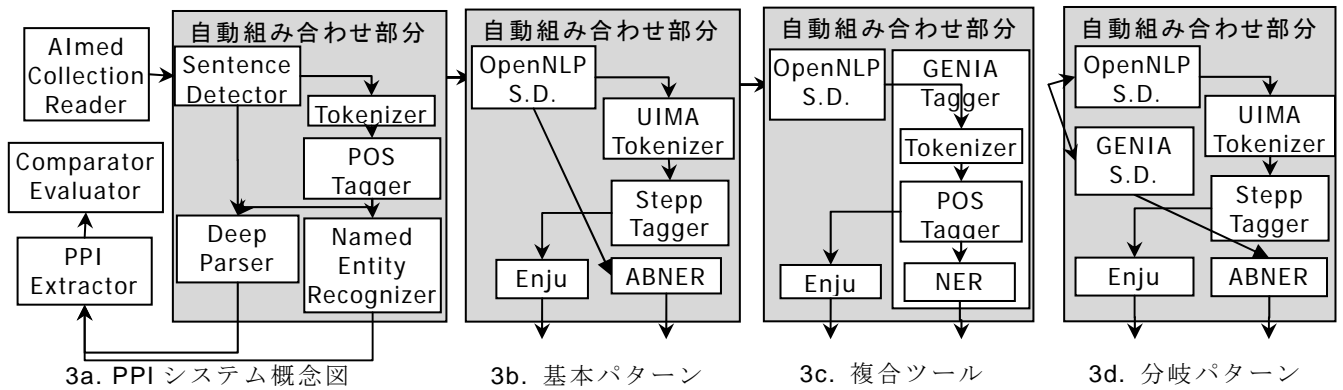


図 3. 自動組み合わせ比較の例

3.2. 自動組み合わせ比較と Type System

自然言語処理タスクは、より細かいコンポーネントの組み合わせで実現できることが多い。図 3 は、生物学論文テキストからタンパク質間相互作用 (PPI, Protein-Protein Interaction) を記述する部分を自動抽出するシステムの例である。我々の PPI システムは、タンパク質の記述を抽出する NER および HPSG 構文パーザの出力を必要とする。この出力を得るために、さらに文検出器 (Sentence Detector)、トークナイザ、品詞タガーが必要である (図 3a)。これら種々のツールについて、それぞれ多数のものを用意 (たとえば、品詞検出器を 4 種類など) できれば、最終的な組み合わせはさらに多くなる。

しかし実際には、図 3b のような単純な組み合わせだけでなく、単一のツールが複合的な機能を持つ場合がある (図 3c)。また、異なるツールが同種の入力を要求する場合、それぞれの入力に異なるツールを組み合わせの方がよい可能性がある (図 3d)。

各ツールがどのような入出力をもつかは、UIMA の Capability を用いて表せる。ただし、type system が異なる入出力を type として表現できるよう、設計されている必要がある。

U-Compare では、type system を適切に設計した上で、入出力 Capability からありうる組み合わせを自動計算し、結果を比較するシステムを構築した。

3.3. ユーザビリティとユーザインターフェース

相互運用性が向上し、ツール間の接続が容易になったとしても、そのためのセットアップなどに労力を強いられないよう、ユーザビリティを向上させることも非常に重要である。

まず、Apache UIMA のセットアップでは、本来自動的に設定できると思われる作業をユーザが行わなければならない。U-Compare では関連パッケージをすべて Web 配布とした上で、Java Web Start を用いて、Web

ブラウザでのワンクリックでシステムが起動⁴するようにした。これにより、Java や UIMA などのバージョンミスマッチも防ぐことができる。

U-Compare による相互運用性は、コンポーネント間の処理順序さえ指定すればすべてを自動的に実行可能にしてくれる。そこで、ドラッグ&ドロップで処理順序を指定できるユーザインターフェースを構築した (図 4)。ブラウザ上でクリックすると、まずこの画面が立ち上がる。画面右側のコンポーネント一覧からドラッグ&ドロップで実行フローを作成するか、定義済みの実行フローを指定した上で実行ボタンを押せば、指定したフローが実行される。

実行したフローの結果が比較可能な場合、精度等の統計的な比較結果が表示される (図 5 左側)。また、具体的なデータを視覚的に表示できる (図 5 右側)。

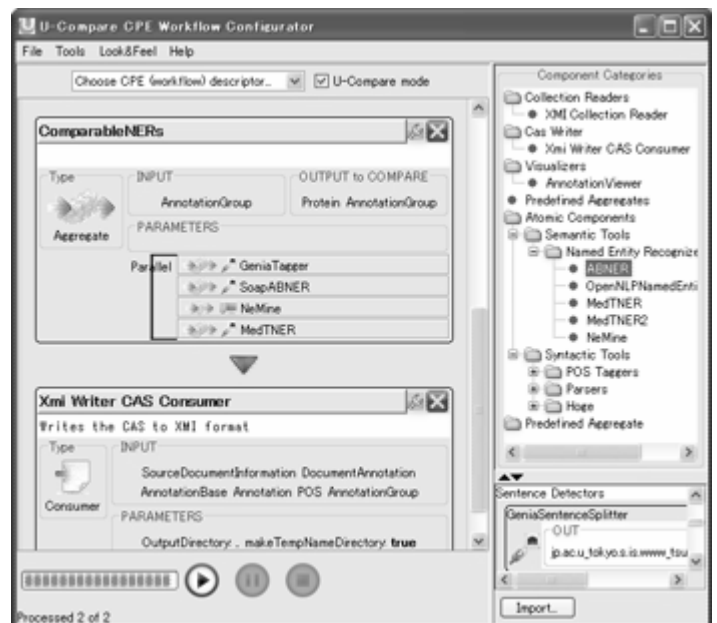


図 4. 実行フローマネージャー画面

⁴ Java 6 実行環境が必要。

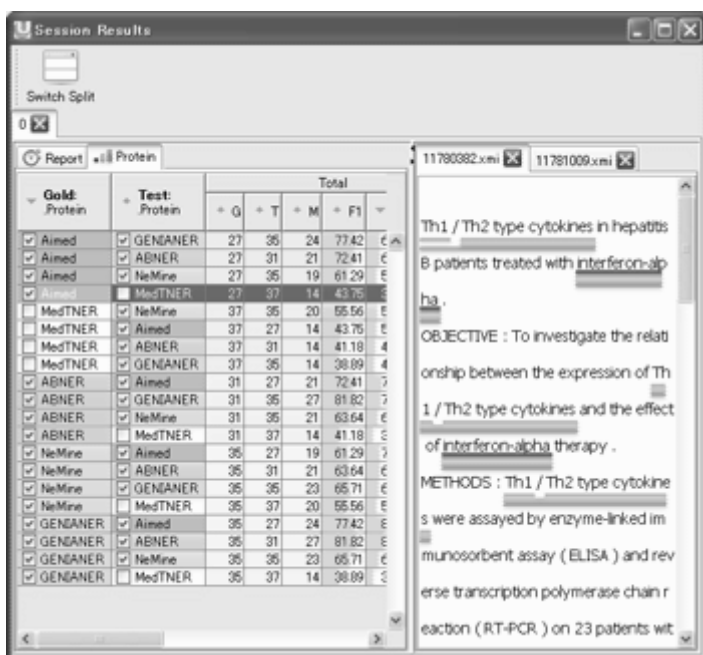


図 5. 処理結果表示画面

3.4. 対応コンポーネントと Web サービス

ここまで述べてきたシステムは UIMA に準拠しており、U-Compare 提供の type system と互換性があれば、任意の UIMA コンポーネントを利用できる。U-Compare では、初期リリースとして 20-30 程度のコンポーネントの提供を予定している(表 1)が、ユーザは自由にコンポーネントを追加して実行することができる。

また、提供されるコンポーネントには Web サービスも多く含まれる。処理の重いツールや、特定の OS でしか実行できないツール、あるいはライセンスの関係で外部公開できないツールでも、Web サービスであれば利用しやすい。

4. おわりに

UIMA は相互運用性を提供する優れたフレームワークであるが、不足点も多く存在する。国際共同プロジェクト U-Compare では各種 UIMA 準拠ツールとともに、sharable type system や自動組み合わせ比較機能、ユーザビリティを向上させる様々なツール群を提供している。

今後の課題としてはまず、構文処理器やコーパスリーダーなど、引き続き対応ツールを増やしていくことが挙げられる。その上で、機械学習ツールを統合し、学習とテストが行えるようデータセットを扱うシステムの追加を考えている。また、Web サービス化が用意であることを生かして、クラスタシステムでの展開とその透過的な利用を計画している。

謝辞

U-Compare システムの構築に当たっては、U-Compare メンバーである、コロラド大学 Computational Pharmacology センター、英国国立テキストマイニングセンター(NaCTeM)、および Luke McCrohon 氏はじめ東京大学辻井研究室のメンバーより多大な協力を頂いたことに心より感謝申し上げます。

また、本研究は、文部科学省科学研究費補助金特別推進研究「高度言語理解のための意味・知識処理の基盤技術に関する研究」、および文部科学省ゲノムネットワークプロジェクトの助成を受けたものである。

表 1. U-Compare 初期提供ツール一覧
(主に東京大学提供のものを記した)

コーパスリーダー	
AimedCollectionReader	MEDLINE 中の 225 アブストラクトに Sentence、Protein,Protein-Protein Interaction をタグ付け [11]
GENIACollectionReader	Pubmed アブストラクトに Protein や Gene,Protein-Protein Interaction をタグ付け [12]
文検出器	
GENIASentenceDetector	GENIA コーパスで訓練された文境界検出器
OpenNLPSentenceDetector	OpenNLP の文検出器を Apache UIMA がラップ
UIMASentenceDetector	Apache UIMA 提供の文検出器
トークナイザ	
GENIATagger	WSJ, GENIA, PennBioIE コーパスで訓練 [13]
OpenNLPTokenizer	OpenNLP のトークナイザを Apache UIMA がラップ
UIMATokenizer	Apache UIMA 提供のトークナイザ
品詞タガー	
GENIATagger	WSJ, GENIA, PennBioIE コーパスで訓練
SteppTagger	確率モデルに基づき WSJ, GENIA, PennBioIE コーパスで訓練、生物学テキスト

	に最適化
OpenNLPTagger	OpenNLP の品詞タグを Apache UIMA がラップ
構文解析器	
Enju	述語項構造や句構造を出力する、Penn Treebank[14] WSJ で訓練された HPSG パーザ[15]
OpenNLPParser	OpenNLP の CFG パーザを Apache UIMA がラップ
Named Entity Recognizer	
ABNER	CRF により NLPBA と BioCreative コーパスで訓練。文検出とトークナイズも一括して行う[16]
GENIATagger	Maximum Entropy[17] を用いて、GENIA コーパスから抽出した NLPBA データセット[18]で訓練。
NeMine	CRF を Genia Corpus/JNLPBA-2004 shared task data で訓練したもので、辞書として BioThesaurus を使用。NaCTeM 作成
MedTNER	JNLPBA データで訓練。NE は MaxEnt classifier で曖昧性解消された Uniprot エントリをもつ。

文 献

- [1] D. Ferrucci, A. Lally, D. Gruhl, E. Epstein, M. Schor, J. W. Murdock, et al. "Towards an Interoperability Standard for Text and Multi-Modal Analytics." IBM Research Report, 2006.
- [2] OASIS UIMA TC. <http://www.oasis-open.org/committees/uima/>.
- [3] Apache UIMA. <http://incubator.apache.org/uima/>.
- [4] UIMA Component Repository. Carnegie Mellon University, <http://uima.lti.cs.cmu.edu/>.
- [5] BioNLP UIMA Component Repository. Center for Computational Pharmacology at the University of Colorado Health Sciences Center, <http://bionlp-uima.sourceforge.net/>.
- [6] Jena University Language & Information Engineering (JULIE) Lab. <http://www.julielab.de/>.
- [7] Tsujii Lab's UIMA Component Repository. Tsujii Laboratory, the University of Tokyo, <http://www-tsujii.is.s.u-tokyo.ac.jp/uima/>.
- [8] Y. Kano, N. Nguyen, R. Sætre, K. Yoshida, Y. Miyao, Y. Tsuruoka, et al. "Filling the gaps between tools and users: a tool comparator, using protein-protein interaction as an example." Proc. Pacific Symposium on Biocomputing (PSB), pp.616-27, Hawaii, USA, January 2008.
- [9] Y. Kano, N. Nguyen, R. Sætre, K. Yoshida, K. Fukamachi, Y. Miyao, et al. "Towards Data And Goal Oriented Analysis: Tool Inter-Operability And Combinatorial Comparison." Proc. 3rd International Joint Conference on Natural Language Processing (IJCNLP), pp.859-64, Hyderabad, India, January 2008.
- [10] U-Compare: share and compare tools with UIMA. <http://u-compare.org/>.
- [11] A. Moschitti. "Making Tree Kernels Practical for Natural Language Learning." Proc. Eleventh International Conference on European Association for Computational Linguistics, Trento, Italy 2006.
- [12] J.-D. Kim, T. Ohta, Y. Teteisi, and J. i. Tsujii. "GENIA corpus - a semantically annotated corpus for bio-textmining." Bioinformatics 19, no. suppl. 1: i180--i82 2003.
- [13] Y. Tsuruoka, Y. Tateishi, J. D. Kim, T. Ohta, J. McNaught, S. Ananiadou, et al. "Developing a robust part-of-speech tagger for biomedical text." In Advances in Informatics, Proceedings, 382-92. Berlin: Springer-Verlag Berlin, 2005.
- [14] M. Marcus, G. Kim, M. A. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, et al. "The Penn Treebank: annotating predicate argument structure." Proc. the workshop on Human Language Technology, Plainsboro, NJ 1994.
- [15] Y. Miyao, and J. Tsujii. "Feature forest models for probabilistic HPSG parsing." Computational Linguistics 34, no. 1: 35-80 2008.
- [16] B. Settles. "ABNER: an open source tool for automatically tagging genes, proteins and other entity names in text." Bioinformatics 21, no. 14: 3191-92 2005.
- [17] A. L. Berger, V. J. DellaPietra, and S. A. DellaPietra. "A maximum entropy approach to natural language processing." Computational Linguistics 22, no. 1: 39-71 1996.
- [18] J.-D. Kim, T. Ohta, Y. Tsuruoka, Y. Tateisi, and N. Collier. "Introduction to the Bio-Entity Recognition Task at JNLPBA." Proc. International Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA-04), pp.70--75, Geneva, Switzerland 2004.