Corpus Architecture*

1. Introduction

The architecture chosen for a certain corpus refers to the conceptual division of different types of objects contained in a corpus, such as texts, annotations and metadata, and the data model containing these objects, e.g. using trees or graphs to connect (parts of) words or documents, and the types of analyses one can apply to each object. This chapter presents some of the key characteristics distinguishing different corpus architectures. The focus is on abstract data models and the ways in which they are realized in concrete formats for corpus representation, as well as consequences for the usability of the resulting corpora.

The overview of fundamental notions in Section 2 is divided into three major sections and begins with an analysis of issues in corpus macro structure, such as dividing corpora into subcorpora, attaching metadata and alignment in parallel corpora. The discussion then moves on to detailed issues of document structure, looking at different types of primary data, such as textual data, transcribed dialogue with or without multiple overlapping speakers, and multimodal data. Although spoken language is considered 'primary' in many senses, corpus architectures usually treat aligned audio/video information (A/V for short) as a type of annotation, and this can have consequences for corpus architecture. As we will explore below, notions such as adjacent tokens, overlapping data, and multiple or conflicting tokenization can arise which have complex effects (see Sauer & Lüdeling 2016). The third subsection completes the overview by discussing architectures for simple textual annotations and more complex annotation graphs (roughly, webs of interconnected analyses), and the ways in which they are encoded. The choice of architecture determines how much information can be expressed, from simple token annotations, such as part of speech (POS) tags, to complex multilayer corpora with conflicting hierarchies encoding syntax, semantics and more in dozens of annotation layers.

Section 3 presents two practical case studies using existing corpora. The first, examining the GUM corpus (Georgetown University Multilayer corpus, Zeldes 2017),

* I would like to thank the editors and two anonymous reviewers for valuable comments on previous versions of this chapter; the usual disclaimers apply.

illustrates aspects of annotation graph modeling, such as positional and structural attributes, span annotations versus hierarchical trees (e.g. syntax trees), and graphs involving pointing relations (for example coreference annotation or discourse relations). These annotation graphs coalesce to form a merged multilayer corpus containing as many as 50 different annotation types applied to each sentence in the corpus. The second study focuses on encoding analyses of non-native language in a learner corpus. Using the MERLIN corpora (Boyd et al. 2014), we discuss using original learner texts and alternative corrected texts, known as target hypotheses, in tandem with error annotations. This creates challenges for the definitions of tokens and other types of word segmentations, with implications for using complex data models with non-standard language.

Section 4 critically outlines some specific formats and methods used by Natural Language Processing (NLP) and manual annotation tools, and compares popular standards in terms of their expressive power, strengths and shortcomings. For building complex corpora, especially in the multilayer annotation graph paradigm, a key tension is discussed between concurrently maintaining multiple, comparatively simple formats for different annotation types, and stand-off XML formats representing 'everything at once'. Section 5 concludes with pointers to useful resources and suggestions for further reading.

2. Fundamentals

2.1 Corpus macro-structure

If a corpus is "a collection of pieces of language that are selected and ordered according to explicit linguistic criteria in order to be used as a sample of the language",[1] then the first component of corpus architecture, before considering analyses within each 'piece', is the organization of the collection of 'pieces of language'. A minimal corpus macro-structure is therefore a single or 'top-level' corpus object, directly containing the 'pieces', which can be referred to as 'documents', as shown on the left in Figure 1. Documents are not necessarily complete texts: for example, they can be samples of *n* tokens (see Biber 1993 on sample size selection), a situation that can arise due to copyright restrictions forbidding full publication of the source text, or due to resource limitations when only a

[1] Definition from EAGLES, the Expert Advisory Group on Language Engineering Standards; see Calzolari & McNaught (1994), and McEnery et al. (2006:4-5) for discussion.

subpart of a longer text can be feasibly annotated in a given project. Documents usually correspond to *contiguous* text taken from some source, with few exceptions.[2] In many corpus search architectures (see Section 4), the definition of the document plays an important role in determining the boundaries of the *search space* for queries: often, if users want to search for certain words 'within 10 words', they intend for the result to come from one document, and would not want to see a search result containing the last word of one text followed by a word from the beginning of an unrelated text. Although this issue is often overlooked, the definition of the document can thus affect search results. For example, in a corpus of the works of Charles Dickens, what are the boundaries of a document? A single book? Or each chapter within each book? While each definition may seem reasonable, they are not identical.

Very often, corpora are constructed according to design criteria which assign documents to different categories (see the chapter on Corpus Compilation). In these cases, the most common corpus macro-structure is the one in the middle of Figure 1: a tree of subcorpora, each containing documents. Subcorpora can be arranged hierarchically, for example a corpus can have written and spoken subcorpora (e.g. corpora in the International Corpus of English, ICE, Greenbaum 1996) and the latter subcorpus may further contain conversation and broadcast news subcorpora, before reaching actual documents. In more complex designs, shown on the right of Figure 1, criteria cross-classify across documents, meaning that documents belong to several categories at once. This is often achieved by labeling documents with metadata categories, with the intention of creating *dynamic* or *virtual subcorpora*. For example, metadata may be used to classify spoken data as a conversation or monologue, and at the same time as private or public speech. It is then possible to dynamically construct a subcorpus containing all private spoken data, or all monologue data, etc.

[2] One example could be in the case of historical corpora of fragmentary texts, in which a document corresponds to everything we have from a certain work which was originally a contiguous text.

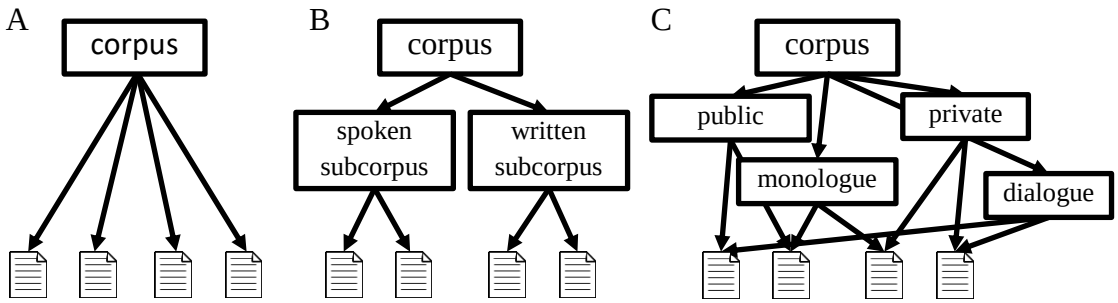

Figure 1. A minimal, flat corpus macro-structure (A), a typical subcorpus tree (B), and a document graph created by cross-classifying metadata (C).

The term *virtual subcorpus* is also used sometimes to refer to querying subsets of earlier queries, i.e. one can dynamically design a 'subcorpus' containing all documents matching an arbitrary query (e.g. the subcorpus of documents containing the word 'snow'), and work further with these documents (see Kupietz et al. 2010: 1852).

While the structures in Figure 1 cover the bulk of corpus resources, some more complex situations deserve special mention. Firstly, in parallel corpora (see Chapter 12 in this volume) containing aligned text, the concept of document is further complicated. Alignment is most often the result of translation corpora, in which each document may exist in more than one language, with alignment either simply at the document level, or more fine-grained forms of alignment, such as section, paragraph, sentence or word alignment (see Romary & Bonhomme 2000 for an extensive discussion). There are also more unusual types of alignment, such as partial alignment (e.g. multilingual corpora of parallel Wikipedia articles which are similar in content, but not actual translations, Smith et al. 2010), corpora aligning editing differences (e.g. corpora of aligned draft revisions, Lee et al. 2015) or corpora containing non-native texts next to aligned target hypotheses of what native annotators believe a non-native speaker is trying to say in the standard target language (see Reznicek et al. 2013). All of these situations complicate the notion of a tree-like graph with simple documents as leaves: in such cases, the leaves themselves may have a complex macro-structure.

2.2 Primary data and text representation

As collections of textual data (in the broad sense, whether written, or transcribed from speech, see Wichmann 2008, or even multimodal corpora of sign language utterances, see

Crasborn & Sloetjes 2008, Schembri et al. 2013), the most fundamental concern in modelling corpora is how text is represented within each document. I will refer to the text being represented as the 'primary data'.[3] While this may seem uncomplicated, it is actually a substantial challenge in many cases. In the first instance, corpus architectures differ in whether or not, or to what extent, they preserve features of the original source data. One of the most frequent violations of primary data integrity in written corpora is white-space preservation. Consider the following example, as formatted, where underscores mark otherwise invisible spaces, and the arrow indicates a tab symbol:

(1) *Mark agreed. This was, then, the end._ _*
|-->But I cannot accept it._ _

Early corpus architectures were aimed at capturing and separating word form tokens, using spaces between token units, often followed by a separator and annotations, as in (2), where a separator '/' marks the beginning of a POS tag (see also the chapter on Corpus Annotation).

(2) *Mark/NNP agreed/VBD ./SENT This/DT was/VBD ,/, then/RB ,/, the/DT end/NN ./SENT But/RB I/PRP can/MD not/RB accept/VB it/PRP ./SENT*

For many linguistic research questions, the representation in (2) is adequate, for example for vocabulary studies: one can extract type/token ratios to study vocabulary size in different texts, find vocabulary preferences of certain authors, etc.

However for many other purposes, the loss of information about the original text from (1) is critical. To name but a few examples:

[3] Note that although A/V signals in multimodal corpora logically precede their transcription, corpus architectures usually implement aligned A/V signals as annotations anchored to the transcription using timestamps. In other words, in much the same way as the POS tag 'noun' might apply to the position in the text of a word like 'bread', a recording of this word is also a type of datum that can be thought of as happening at the point in which 'bread' is uttered. In continuous primary data representations (see below), A/V timestamp alignment therefore 'tiles' the text (no span of audio is left without alignment to some token).

- **Tokens with ambiguous spacing:** both ‘can not’ and ‘cannot’ are usually tokenized as two units, but to study variation between these forms, one needs to represent whitespace somehow
- **Training automatic sentence/document/subsection splitters:** Position and number of spaces, as well as presence of tab characters are very strong cues for such programs. For example TextTiling, a classic approach to automatic document segmentation, makes use of tabs as predictors (Hearst 1997).
- **Stylometry and authorship attribution:** even subtle cues found in whitespace can distinguish authors and styles. For example, US authors are much more likely to use double spaces after a sentence final period than UK authors, and specific combinations of whitespace practices can sometimes uniquely identify authors (see Kredens & Coulthard 2012:506-507). Proportion of white space has also been used in authorship and plagiarism detection (Canales et al. 2011).

Whitespace and other features of the original primary data can therefore be important, and some corpus architectures employ formats which preserve and separate the underlying data from processes of tokenization and annotation, often using ‘stand-off’ XML formats. In stand-off formats, different layers of information are stored in separate files using a referencing mechanism which allows us, for example, to leave an original text file unchanged. One can then add e.g. POS annotations in a separate file specifying the character offsets in the text file at which the annotations apply (e.g. marking that a NOUN occurred between characters 4-10; see Tools and Resources for more details).

A second important issue in representing language data is the tokenization itself, which requires detailed guidelines, and is usually executed automatically, possibly with manual correction (see Schmid 2008 for an overview). Although a working definition of ‘tokens’ often equates them with “words, numbers, punctuation marks, parentheses, quotation marks, and similar entities” (Schmid 2008:527), a more precise definition of tokens is simply “the smallest unit of a corpus” (Krause et al. 2012:2), where units can also be smaller than a word, e.g. in a corpus treating each syllable as a token. In other words, tokens are minimal, indivisible or ‘atomic’ units, and any unit to which we want to apply annotations cannot be smaller than a token (see Section 3.2).

In English, word forms and tokens usually coincide, and tokenization is closely related to prevalent part of speech tagging guidelines (the Penn tag set, Santorini 1990 and CLAWS, Garside et al. 1987, both ultimately going back to the Brown tag set, Kučera & Francis 1967). However, modals, negations and other items which sometimes appear as clitics are normally tokenized apart, as in the clitics *'ll* and *n't* in (3) and (4). These are represented as separate in the 'tok' (token) rows of Figure 2, but are fused on the 'wf' (word form) level. In (3), separating the clitic *'ll* allows us to tag it as a modal on the 'pos' layer (MD), just like a normal *will*. The other half of the orthographic sequence *I'll* is retained unproblematically as *I*. In (4), by contrast, separating the negation *n't* produces a segment *wo*, which is not a 'normal' word in English, but is nevertheless tagged as a modal.

(3) *I'll do it*

(4) *I won't do it then*

In order to make all instances of the lexical item *will* findable, some corpora rely on lemmatization (the lemma of all of these is *will*), while other corpora use explicit normalization. This distinction becomes more crucial in corpora with non-standard orthography, as in example (5), featuring the contraction *I'm a* (frequent in, but not limited to African American Vernacular English, Green 2002:196).

(5) *I'm a do it* (i.e. I'm going to do it)

(3) wf	**I'll**		**do**	**it**		
norm	I	will	do	it		
pos	PRP	MD	VB	PRP		
tok	**I**	**'ll**	**do**	**it**		
(4) wf	**I**	**won't**		**do**	**it**	**then**
norm	I	will	not	do	it	then
pos	PRP	MD	RB	VB	PRP	RB
tok	**I**	**wo**	**n't**	**do**	**it**	**then**

(5) wf	**I'm**		**a**		**do**	**it**
norm	I	am	going	to	do	it
pos	PRP	VBP	VBG	TO	VB	PRP
tok	**I**	**'m**	**a**		**do**	**it**

Figure 2. Tokenization, normalization and POS tags for word forms in (3)-(5).

This last example clearly shows that space-delimited orthographic borders, tokenization, and annotations at the word form level may not coincide. To do justice to examples such as (5), a corpus architecture must be capable of mapping word forms and annotations to any number of tokens, in the sense of minimal units. In some cases these tokens may even be empty, as in the position following *a* in the 'tok' layer for (5) – what matters is not necessarily that 'tok' contains some segmentation of the text in 'wf', but rather that the positions and borders that are required for the annotation table are delimited correctly in order to allow the interpretation of *a* as corresponding to the 'norm' sequence *going* (tagged VBG) and *to* (tagged TO), assuming this is the desired analysis.[4]

For multimodal data in which speakers may overlap, the situation is even more complex and an architecture completely separating the concepts of tokens as minimal units and word forms becomes necessary. An example is shown in Figure 3.

spkA	I	see			but	actually		I
posA	PRP	VBP			PRP	RB		PRP
spkB							you	know
posB							PRP	VBP
events		[phone rings]						
time	00:03.1	00:04	00:04.2	00:05.6	00:07	00:07.5	00:08	00:08.1

Figure 3. Multiple layers for dialog data with a minimally granular timeline.

The example shows several issues: towards the end, two speakers overlap with word forms that only partially occur at the same time, meaning that borders are needed

[4] Some architectures go even further and use an 'empty' token layer, using tokens solely as ordered positions or time-line markers, not containing text (e.g. the RIDGES corpus, Odebrecht et al. 2016, or REM, Klein & Dipper 2016). In such cases, tools manipulating the data can recover the covered text for each position from an aligned primary text.

corresponding to these offsets; in the middle of the excerpt, there is a moment of silence, which has a certain duration; and finally, there is an extra linguistic event (a phone ringing) which takes place in part during speaker A's dialogue, and in part during the silence between speech acts.

An architecture using the necessary minimal units can still represent even this degree of complexity, provided that one draws the correct borders at the minimal transitions between events, and add higher level spans for each layer of information. In cases like these, the concept of minimal token is essentially tantamount to timeline indices, and if these have explicit references to time (as in the seconds and milliseconds in the 'time' layer of Figure 3), then they can be used for A/V signal alignment as well. An architecture of this kind is used by concrete speech corpus transcription tools such as ELAN (Brugman & Russel 2004) or EXMARaLDA (Schmidt & Wörner 2009), but can more generally be thought of as an annotation graph (see the next section).

A final consideration in cases such as these is the anchoring or coupling of specific layers of information in the data model: in the example above, the two 'pos' layers belong to the different speakers. A user searching for all word forms coinciding with a verbal tag in the corpus would be very surprised to find the word *I*, which might be found if all VBP tags coinciding with a word form are retrieved (since the second *I* overlaps with the other speaker's word *know*). What is meant in such situations is to only look at combinations of POS and word form information from either speaker A or speaker B. In other situations, however, one might want to look at any layers containing some speaker (e.g. search for anyone saying *um*), in which case some means of capturing the notion of 'any transcription layer' is required. These concepts of connecting annotation layers (posA belongs to spkA) and applying multiple segmentations to the data will be discussed below in the context of graph models for corpus annotations.

2.3 Data models for document annotations

The central concern of annotations is 'adding interpretative, linguistic information to an electronic corpus' (Leech 1997:2), such as adding POS tags to word forms (see Chapter 3 on Corpus Annotation). However, as we have seen, one may also want to express relationships between annotations, grouping together multiple units into larger spans, building structures on top of these, and annotating them in turn. For example, multiple

minimal tokens annotated as morphemes may be grouped together to delineate a complex word form, several such word forms may be joined into phrases or sentences, and each of these may carry annotations as well. Additionally, some annotations 'belong together' in some sense, for example by relating to the same speaker in a dialogue. If a document contains these kinds of data, the resulting structure is then no longer a flat table such as Figure 2, but rather a graph with explicit hierarchical connections. For planning and choosing a fitting corpus architecture, it is important to understand the components of annotation graphs at an abstract level, since even if individual XML formats under consideration for a corpus vary substantially (see Section 5.1), at an underlying level, the most important factor is which elements of an annotation graph they can or cannot represent.

At its most general formulation, a graph is just a collection of nodes connected by edges: for example an ordered sequence of words, each word connected to the next, with some added nodes connected to multiple words (e.g. a sentence node grouping some words, or smaller phrase nodes). Often these nodes and edges will be annotated with labels, which usually have a category name and a value (e.g. POS=NOUN); in some complex architectures, annotations can potentially include more complex data types, such as hierarchical feature structures (see ISO24612). Additionally, some data models add grouping mechanisms to annotation graphs, often referred to as 'annotation layers', which can be used to lump together annotations that are somehow related.[5]

Given the basic building blocks 'nodes', 'edges', 'annotations' and 'layers', there are many different constraints that can be imposed on the combinations of these elements. Some data models allow us to attach annotations only to nodes, or also to edges; some data models even allow annotations of annotations (e.g. Dipper 2005), which opens up the possibility of annotation sub-graphs expressing, for example, provenance (i.e. who or what created an annotation and when, see Eckart de Castilho et al. 2017) or certainty of

[5] In some formats, XML namespaces form layers to distinguish annotations from different inventories, such as tags from the TEI vocabulary (Text Encoding Initiative, http://www.tei-c.org/) versus corpus specific tags (see Höder 2012 for an example). A formal concept of layers to group annotations is provided in the Salt data model (Zipser & Romary 2010), and UIMA Feature Structure Types in the NLP tool-chain DKPro (Eckart de Castilho & Gurevyich 2014). NLP tool chain components are often thought of as creating implicit layers (e.g. a parser component adds a syntactic annotation layer), see e.g. GATE Processing Resources or CREOLE Modules in GATE (Cunningham et al. 1997), Annotators components in CoreNLP (Manning et al. 2014) or WebLicht Components (Hinrichs et al. 2010).

annotations (e.g. an ‘uncertain’ label, or numerical likelihood estimate of annotation accuracy). Another annotation model constraint is whether multiple instances of the same annotation in the same position are allowed (e.g. conflicting versions of the same annotation, such as multiple POS tags or even syntax trees, see Kountz et al. 2008). This can be relevant not only for fine-grained manual annotations, but also for the application and comparison of multiple automatic tools (several POS taggers, parsers, etc.). Layers too can have different constraints, including whether layers can be applied only to nodes, or also to edges and annotations, and whether layer-element mapping is 1:1 or whether an element can belong to multiple layers. Search engines sometimes organize visualizations by layers, i.e. using a dedicated syntax tree visualization for a ‘syntax’ layer, and other modules for annotations in other layers.

Basic annotation graphs, such as syntactically annotated treebanks, can be described in simple inline formats. However, as the corpus architecture grows more complex or ‘multilayered’, the pressure to separate annotations into different files and/or more complex formats grows. To see why, one can consider the Penn Treebank’s (Marcus et al. 1993) bracketing format, which was developed to encode constituent syntax trees. The format uses special symbols to record not only the primary text, but also empty categories, such as *pro* (for dropped subject pronouns), *PRO* (for infinitive subjects), traces (for postulated movement), and more. In the following tree excerpt from the Wall Street Journal portion of the Penn Treebank, there are two ‘empty categories’, at the two next to last tokens: a zero ‘0’ tagged as -NONE- standing in for an omitted *that* (i.e. *researchers said *that**), and a trace ‘*T*-2’, indicating that a clause has been fronted (i.e. the text is “*crocidolite is … resilient …, researchers said*”, which can be considered to be fronted from a form such as “*researchers said the crocidolite…*”):

```
( (S
    (S-TPC-2
      (NP-SBJ
…
        (NP (NN crocidolite) )
        (, ,) )
      (VP (VBZ is)
```

```
        (ADJP-PRD (RB unusually) (JJ resilient) )
…
    (, ,)
    (NP-SBJ (NNS researchers) )
    (VP (VBD said)
      (SBAR (-NONE- 0)
        (S (-NONE- *T*-2) )))
    (. .) ))
```

This syntax tree defines a hierarchically nested annotation graph, with vertices (*V*) corresponding to the tokens and bracketing nodes, and annotations corresponding to parts of speech and syntactic category labels (NP, VP etc.). However much of the information is rather implicit; the edges of the tree are marked by nested brackets: the NP dominates the noun ‘crocidolite’, etc. Annotations are pure value labels (VBD, VP etc.), and one must infer readings for their keys (POS, phrase category). Another ‘edge’ represented by co-indexing the trace with its location at S-TPC-2 depends on our understanding that *T*-2 is not just a normal token (marked only by a special POS tag -NONE-). This is especially crucial for the dropped ‘that’, since the number 0 can also appear as a literal token, for example in the following case, also from the Wall Street Journal section of the Penn Treebank:

```
(NP
  (NP (DT a) (NN vote) )
  (PP (IN of)
      (NP
        (NP (CD 89) )
        (PP (TO to)
            (NP (CD 0) )))))
```

At the very latest, once information unrelated to the syntax tree is to be added to the corpus, such as temporal annotation, coreference resolution or named entity tags, multiple annotation files will be needed. In fact, the OntoNotes corpus (Hovy et al. 2006), which contains parts of the Penn Treebank extended with multiple additional layers, is indeed serialized in multiple files for each document, expressing unrelated or only

loosely connected layers of annotation. A corpus containing unrelated layers in this fashion is often referred to as a 'multilayer' corpus, and data models and technology for such corpora are an active area of research (see Lüdeling et al. 2005, Burchardt et al. 2008, Zeldes 2017 and forthcoming).

Because of the complexity inherent in annotation graphs, complex tools are often needed to annotate and represent multilayer data, and the choice of search and visualization tools with corresponding support becomes more limited (see Tools and Resources). In the case of formats for data representation, the situation is somewhat less critical, since, as already noted, different types of information can be saved in separate files. This also extends into the choice of annotation tools, as one can use separate tools, for example to annotate syntax trees, typographical properties of source documents, or discourse annotations. The greater challenge begins once these representations need to be merged. This is often only possible if tools ensuring consistency across layers are developed (e.g. the underlying text, and perhaps also tokenization must be kept consistent across tools and formats).

As a merged representation for complex architectures, stand-off XML formats are often used (see Section 4), and Application Programmatic Interfaces (APIs) are often developed in tandem with such corpora to implement validation, merging and conversion of separate representations of the same data (for example, the ANC Tool, used to convert data in the American National Corpus and its richly annotated subcorpus, MASC, Ide et al. 2010). For search and visualization of multilayer architectures, either a complex tool can be used, such as ANNIS (Krause & Zeldes 2016, see Section 5.1), or a combination of tools is used for each layer. For example in the TXM text mining platform, Heiden (2010) proposes to use a web interface to query the Corpus Workbench (Christ 1994) for 'flat' annotations, TigerSearch (Lezius 2002) for syntax trees, and XQuery for hierarchical XML. The advantage of this approach is that it can use off-the-shelf tools for a variety of annotation types, and that it can potentially scale better for large corpora, since each tool has only a limited share of the workload. The disadvantage is that a data model merging results from all components can only be generated after query retrieval has occurred in each component. This prevents complex searches across all annotation layers: for example, it is impossible to find sentences with certain syntactic properties,

such as clefts, which also contain certain XML tags, such as entity annotations denoting persons, and also have relational edges with components of other sentences, such as coreference with a preceding or following entity annotation. These kinds of combinations can be important for example for studying the interplay between syntax and semantics, especially at the levels of discourse and pragmatics.

3. Case studies

3.1 The GUM corpus

The Georgetown University Multilayer corpus (GUM, Zeldes 2017), is a freely available corpus of English Web genres, created using 'class-sourcing' as part of the Linguistics curriculum at Georgetown University. The corpus, which is expanded every year and currently contains over 64,000 tokens, is collected from four open access sources: Wikinews news reports, Wikimedia interviews, wikiHow how-to guides and Wikivoyage travel guides. Its architecture can therefore be considered to follow the common tree-style macro-structure with four subcorpora, each containing simple, unaligned documents. The complexity of the corpus architecture results from its annotations: as the data is collected, student annotators iteratively apply a large number of annotation schemes to their data using different formats and tools, including document structure in TEI XML, POS tagging, syntactic parsing, entity and coreference annotations and discourse parses in Rhetorical Structure Theory. The complete corpus covers over 50 annotation types (see http://corpling.uis.georgetown.edu/gum/). A single tokenized word in GUM therefore often carries an annotation graph of dozens of nodes and annotations, illustrated using only two tokens from the corpus in Figure 4, which shows the two tokens *I know*.

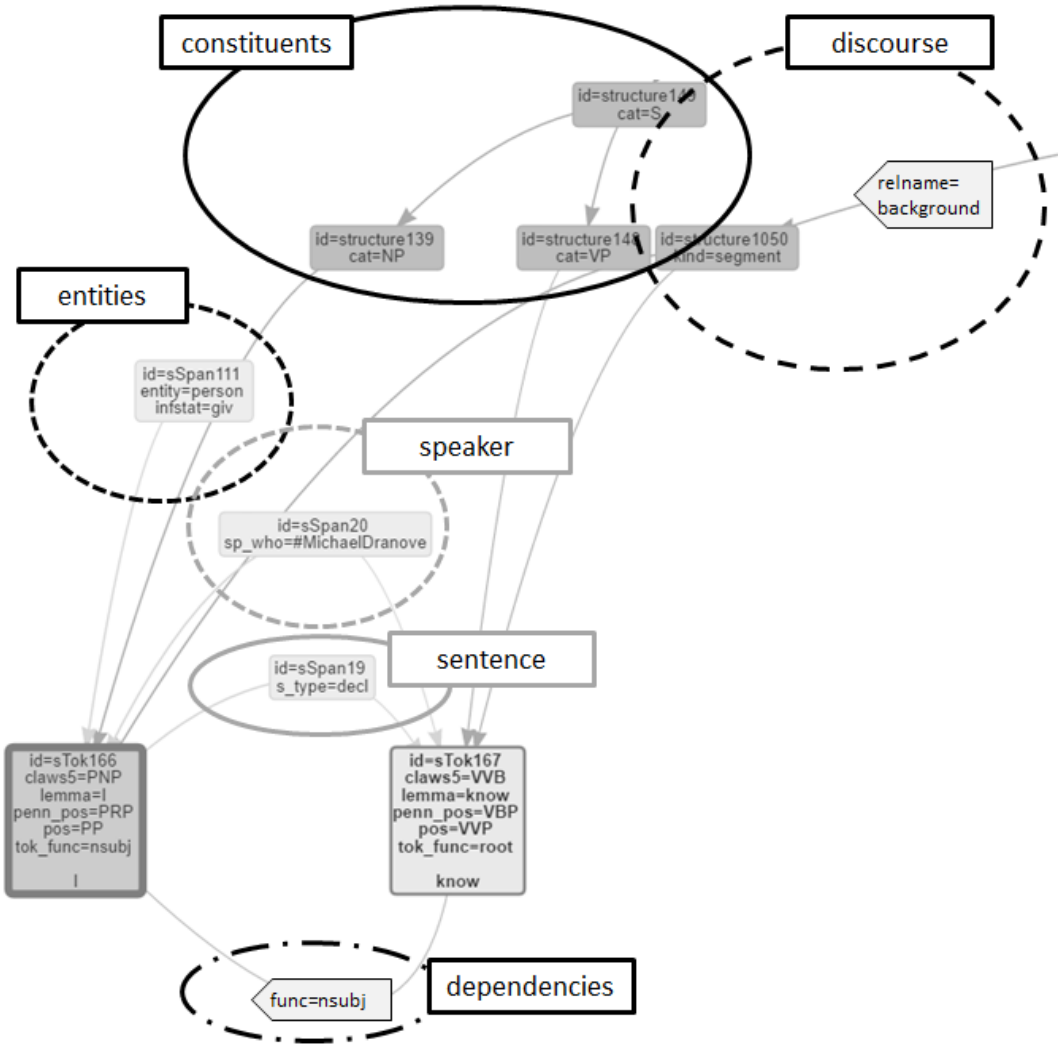

Figure 4. Annotation graph for the tokens *I know* in an interview from GUM.

At an abstract level, the boxes in Figure 4 represent general graph nodes from the set *V*. The two tokens in the shaded boxes towards the bottom of the image are somewhat special nodes in that they carry both a variety of annotations (part of the set *A*) and references to primary text data (*I* and *know*). Their annotations including three distinct POS tags based on different tag sets, as well as the lemma and grammatical function annotation. These token nodes also function as anchors for the remaining nodes in the graph: every other node in the figure is attached directly or indirectly to the tokens via edges. Layers are represented by distinct ovals; in this case, the tokens have not been placed in any layer, but all other nodes and edges belong to exactly one layer. For example, there is a dependency edge at the bottom of the figure connecting the two

tokens and carrying a label (func=nsubj, since *I* is the nominal subject of *know*), belonging to a layer of 'dependencies'. The single node in the layer 'sentence' above the tokens is annotated as s_type=decl (declarative sentence), and is attached to both tokens, but the edges attaching it are unannotated (no labels). Finally, some layers, such as the constituent layer, contain a complex subgraph: an NP node is attached to the token *I* in the 'constituents' layers, and a VP node is attached to *know*, and together they attach to the *S* node denoting the clause. Similarly, the discourse layer, of which we only see one incoming edge, is the entry point into the discourse annotation part of the graph, which places multiple tokens in segments, and then constructs a sub-graph made of sentences and clauses based on Rhetorical Structure Theory (RST, Mann & Thompson 1988). The edge is annotated as 'background', indicating this clause gives background information for some other clause.

Note that it is the corpus designer's decision which elements are grouped in a layer. For example, the constituent annotation *S* for the clause has a similar meaning to the sentence annotation in the 'sentence' layer, but these have been modeled as separate. As a result, it is at least technically possible for the corpus to have conflicting constituent trees and sentence span borders. If these layers are generated by separate automatic or manual annotation tools, then such conflicts are in fact likely to occur over the course of the corpus. Similarly, a speaker annotation ('sp_who') is attached to both tokens, as is the sentence annotation, but it is conceivable that these may conflict hierarchically: a single sentence annotation may theoretically cover tokens belonging to different speakers, which may or may not be desirable (e.g. for annotating one speaker completing another's sentence). This data models allows for completely independent annotation layers, united only by joint reference to the same primary text.

3.2 The MERLIN Corpus

The MERLIN project (Multilingual Platform for European Reference Levels: Interlanguage Exploration in Context, Boyd et al. 2014) makes three learner corpora available in the target languages Czech, German and Italian, which are richly annotated and follow a comparable architecture to allow for cross-target language and native language comparisons. The project was conceived to collect, study and make available learner texts across the levels of the Common European Framework of Reference for

Languages (CEFR), which places language learners at levels ranging from A1 (also called 'Breakthrough', the most basic level) to C2 ('Mastery'). Although these levels are commonly used in language education and studies of second language acquisition, learners often have little or no possibility to find texts coming from these levels. The MERLIN corpora fill this gap by making texts at the A1-C1 levels publically available in the three target languages above.

To see how the MERLIN corpora take advantage of their architecture in order to expose learner data across levels we must first consider how users may want to access the data, and what the nature of the underlying primary textual data is. On one level, researchers, language instructors and other users would like to be able to search through learner data directly: the base text is, trivially, whatever a learner may have written. However, at the same time the problems discussed in Section 2.2 make searching through non-native data, which potentially contains many errors,[6] non-trivial. For example, the excerpt from one Italian text in (6) contains multiple errors where articles should be combined with prepositions: once, *da* 'from' is used without an article in *da mattina* 'from (the) morning' for *dalla* 'from the (feminine)', and once, the form *da* is used instead of *dal* 'from the (masculine)'. The data comes from a Hungarian native speaker, rated at an overall CEFR ranking of B2, as indicated by document metadata in the corpus.

(6) *Da mattina al pomerrigio? Da prossima mese posso lavorare?*
from morning to.the afternoon ? from next month can.1.SG work ?
'From morning to the afternoon? From next month I can work?'

This data is invaluable to learners and educators interested in article errors. However users interested in finding all usages of *da* in the L2 data will not be able to distinguish correct cases of *da* from cases that should have *dal* or *dalla*. At the same time, less obvious errors may render some word forms virtually unfindable. For example, the word *pomeriggio* 'afternoon' is misspelled in this example, and should read *pomerrigio* (the 'r'

[6] This is not to say that native data does not contain errors from a normative perspective, and indeed some corpora, such as GUM in Section 3.1, do in fact annotate native data for errors.

should be double, the 'g' should not be). As a result, users who cannot guess the actual spelling of words they are interested in will not be able to find such cases.

In order to address this, MERLIN includes layers of target hypotheses (TH, see Reznicek et al. 2013). These provide corrected versions of the learner texts: At a minimum, all subcorpora include a span annotation called TH1, which gives a minimally grammatical version of the learner utterance, correcting only a much as necessary to make the sentence error-free, but without improving style or correcting for meaning.[7] Figure 5 shows the learner utterances on the 'learner' layer, while the TH1 layer shows the minimal correction: preposition+article forms have been altered, and a word-order error in the second utterance has been corrected (the sentence should begin *Posso lavorare* 'can I work'). The layer TH1Diff further notes where word form changes have occurred (the value 'CHA'), or where material has been moved, using 'MOVS' (moved, source) and 'MOVT' (moved, target).[8] These 'difference tags' allow users to find all cases of discrepancies between the learner text and TH1 without specifying the exact forms being searched for.

learner	Da	mattina	al	pomerrigio	?			Da	prossima	mese	posso	lavorare	?	
TH1	Dalla	mattina	al	pomeriggio	?	Posso	lavorare	dal	prossimo	mese			?	
TH1Diff	CHA			CHA		MOVT	MOVT	CHA	CHA		MOVS	MOVS		
EA_category				O_Graph				G_Wo						
EA_category	G_Art								G_Morphol_Wrong					
G_Art_type	o							o						
G_Morphol_Wrong_type									gend					
G_Wo_type								womaincl						
O_Graph_graphgen_act_type				o										
O_Graph_graphgen_act_type				ad										
O_Graph_type				graphgen										

Figure 5. Annotation grid for a learner utterance, with target hypothesis (TH) and error annotations, visualized using ANNIS (see Section 5.1).

One consequence of using a TH layer for the architecture of the corpus is that the data may now in effect have two conflicting tokenizations: on the 'learner' layer, the first

[7] See Reznicek et al. (2012) for the closely related Falko corpus of L2 German, which developed minimal TH annotation guidelines. Like Falko, a subset of MERLIN also includes an 'extended' TH layer, called TH2, on which semantics and style are also corrected. A closely related concept to TH construction which is relevant to historical corpora is that of normalization: non-standard historical spellings can also be normalized to different degrees, and similar questions about the desired level of normalization often arise.

[8] Further tags include 'DEL' for deleted material, and 'INS' for insertions.

'?' and the second 'Da' stand at adjacent token positions; on the TH1 layer, they do not. To make it possible to find '?' followed by 'Da' in this instance, while ignoring or including TH layer gaps, MERLIN's architecture explicitly flags these annotation layers as 'segmentations', allowing a search engine to use either one for the purpose of determining adjacency as well as context display size (i.e. what to show when users request a windows of +/- 5 units).

One shortcoming of TH annotations is that they cannot generalize over common error types which are of interest to users: for example, they do not directly encode a concept of 'article errors'. To remedy this, MERLIN includes a wide range of error annotations, with a major error-annotation category layer (EA_category, e.g. G_Morphol_Wrong for morphological errors), and more fine grained layers, such as G_Morphol_Wrong_type. The latter indicates a 'gender' error on *prossima* 'next (feminine)' in Figure 5, which should read *prossimo* 'next (masculine)' to agree with *mese* 'month'. Note however that the architecture allows multiple conflicting annotations at the same position: two 'EA_category' annotations overlap under *prossima*, indicating the presence of two concurrent errors, and there is no real indication, except for the length of the span, that the 'gender' error is somehow paired with the shorter EA_category annotation. Additionally, the EA layers cannot encode all foreseeable errors of interest: for example, there is no specific category for cases where *da* should be *dal* (but not *dalla*). This type of query can only be addressed using the running TH layer.[9]

Finally it should be noted that both tokens and annotations, including TH layers, can be used as entry points for more complex annotation graphs. In the case of MERLIN, an automatically generated dependency syntax parse layer was added on top of learner layer, as shown in Figure 6.

[9] A more minimal type of TH analysis is also possible, in which only erroneous tokens are given a correction annotation (see e.g. Tenfjord et al. 2006 for a solution using TEI XML). A limitation of this approach is that the TH layer itself cannot be annotated as a complete independent text (e.g. to compare POS tag distributions in the original and TH text), and that gaps of the type seen in Figure 5 cannot be represented.

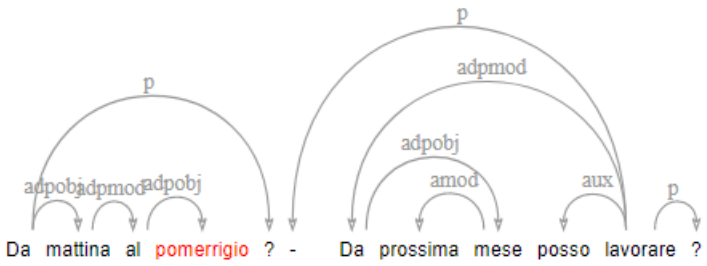

Figure 6. Dependency parse attached to the annotations of example (6) in MERLIN.

If the corpus architecture has successfully expressed all annotations including the parse in a single graph, then it is possible to query syntax trees in conjunction with other layers. For example we can obtain syntactic information, such as the most common grammatical functions and distance between words associated with movements (MOVS/MOVT) across gaps on the TH layer. This would not be possible if TH analysis had been implemented in separate files, without consideration for the alignment of each annotation's structures or the handling of gaps and segmentation conflicts. Similar additional graphs would also be conceivable, for example to link specific MOVS and MOVT locations, but these have not yet been implemented – TH1Diffs are currently expressed as flat annotations whose interconnections are left unexpressed in the data model.

4. Critical assessment and future directions

At the time of writing, corpus practitioners are in the happy position of having a wide range of choices for concrete corpus representation formats and tools. However, few tools or formats can do 'everything', and more often than not, the closer they get to this ideal, the less convenient or optimized they are for any one task. To recap some important considerations in choosing a corpus architecture and a corresponding concrete representation format:

- Is preservation of the exact underlying text (e.g. whitespace preservation) important?
- Are annotations very numerous or involve conflicting spans to the extent that a stand-off format is needed?

- Are annotations arranged in mutually exclusive spans? Are they hierarchically nested? Are discontinuous annotations required?
- Are complex metadata management and subcorpus structure needed, or can this information be saved separately in a simple table?
- Does the data contain A/V signals? If so, are there overlapping speakers in dialogue?
- Is parallel alignment needed, i.e. a parallel corpus?

These questions are important to address, but the answers are not always straightforward. For example, one can represent 'discontinuous' annotations slightly less faithfully by making two annotations with some co-indexed naming mechanism (cf. MOVS and MOVT in Section 3.2). This may be unfaithful to our envisioned data model, but will greatly broaden the range of tools that can be used.

In practice, a large part of the choice of corpus architecture is often dictated by the annotation tools that researchers wish to use, and the properties of their representation formats. Using a more convenient tool and compromising the data model can be the right decision if this compromise does not hurt our ability to approach our research questions or applications. For example, many spoken corpora containing dialogue do not model speaker overlap, instead opting to place overlapping utterances in the order in which they begin. This can be fine for some research questions, for example for a study on word formation in spoken language; but not for others, e.g. for pragmatic studies of speech act interactions in dialogue. Table 1 gives a (non-exhaustive) overview of some popular corpus formats and their coverage in terms of the properties discussed above. A good starting point when looking to choose a format is to use this table or construct a similar one, note supported and unsupported features, and rule out formats that are not capable of representing the desired architectural properties.

	whitespace	standoff	hierarchy	confl. spans	discontinuous	parallel	dialogue overlap	metadata	subcorpora	multimodal

CoNLLU	yes	no	dep[10]	no	no	no	no	no	no	no
CWB	no	no	no	yes	no	yes	no	yes	no	no
Elan	yes	inline	no	yes	no	no	yes	yes	yes	yes
EXMARaLDA	yes	inline	no	yes	no	no	yes	yes	yes	yes
FoLiA	yes	inline	yes	yes	yes	no	no	yes	yes	no
GrAF	yes	yes	yes	yes	yes	no[11]	no[11]	yes	yes	no
PAULA XML	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
PTB	no	no	yes	no	no	no	no	no	no	no
TCF	yes	inline	yes	yes	yes	no	no	yes	no	no
TEI XML	yes	yes[12]	yes	no[12]	no[12]	yes	yes	yes	yes	yes
TigerXML	no	no	yes	no	yes	no[13]	no	yes[13]	yes	no
tiger2	yes	yes	yes	yes	yes	no	no	yes	yes	no
WebAnno	yes	inline	dep[10]	yes	no	no	no	no	no	no

Table 1. Data model properties for a range of open corpus formats.

CoNLLU is a popular format in the family of tab-delimited CoNLL formats, which is used for dependency treebanks in the Universal Dependencies project (http://universaldependencies.org/). It is enriched with 'super-token'-like word forms (i.e. multi-token orthographic word forms such as 'I'm'), open-ended key-value pairs on tokens, and sentence level annotations as key-value pairs. The CWB vertical format (sometimes also called 'TreeTagger format', due to its compatibility with the tagger by Schmid 1994), is an SGML format with one token per line, accompanied by tab-delimited token annotations, and potentially conflicting, but not hierarchically nested element spans. Elan and EXMARaLDA are two popular grid-based annotation tools, which do not necessarily model a token concept, instead opting for unrestricted layers of spans, some of which can be used to transcribe texts, while others express annotations. They offer excellent support of aligned A/V data and model a concept of potentially

[10] The value 'dep' indicates formats with some capacity to express dependency edges between flat units (including, e.g. syntactic dependency or coreference annotation), but without complex node hierarchies.
[11] While GrAF does not explicitly support multiple overlapping speakers or parallel corpora, there are some conceivable ways of representing these using the available graph structure. However I am not aware of any corpus or tool implementing these with GrAF.
[12] Stand-off annotation has been implemented in TEI XML (see Chapter 20.5 of the TEI p5 guidelines, http://www.tei-c.org/) and can cover a wide range of use cases for discontinuous annotations and hierarchy conflicts. However it is not frequently used in the TEI community, and there are some limitations (see Bański 2010 for analysis).
[13] TigerXML itself does not implement parallel alignment, but an extension format known as STAX has been developed for parallel treebanking in the Stockhold TreeAligner (Lundborg et al. 2007). Metadata in TigerXML is limited to a predetermined set of fields, such as 'author', 'date' and 'description'.

multiple speakers, complete with speaker-related metadata, which makes them ideal for dialogue annotation. FoLiA, GrAF and PAULA XML are all forms of graph-based stand-off XML formats, though FoLiA's implementation is actually contained in a single XML file, with document internal references. GrAF has the status of an ISO standard (ISO 24612), and has been used to represent the American National Corpus (https://www.anc.org/). FoLiA has the advantage of offering a complete annotation environment (FLAT, http://flat.science.ru.nl/), though PAULA and GrAF can be edited using multi-format annotation tools such as Atomic (http://corpus-tools.org/atomic/). PAULA is the only format of the three which implements support for parallel corpora and overlapping speakers.

Penn Treebank bracketing (PTB), TigerXML and tiger2 are formats specializing in syntax annotation (treebanks). The PTB format is the most popular way of representing projective constituent trees (no crossing edges) with single node annotations (part of speech or syntactic category). It is highly efficient and readable, but has some limitations (see the 'crocidolite' example above). TigerXML is a more expressive XML format, capable of representing multiple node annotations, crossing edges, edge labels and two distinct types of edges. The tiger2 format (Romary et al. 2015) is an extension of TigerXML, outwardly very similar in syntax, but with unlimited edge typing, metadata, multiple/conflicting graphs per sentence and other more 'graph-like' features. It enjoys an ISO standard status (ISO 24615).

TCF (Hinrichs et al. 2010) is an exchange format used by CLARIN infrastructure, and in particular the WebLicht NLP toolchain. It is highly expressive for a closed set of multilayer annotations, and has built in concepts for tokenization, sentence segmentation, syntax and entity annotation. It is also one of the supported formats of the popular WebAnno online annotation tool (Yimam et al. 2013), which also supports a variety of formats of its own, including its highly expressive UIMA based format (serializable as an 'inline stand-off' XMI format), and a whitespace preserving tab-delimited export, called WebAnno TSV.

An important trend in corpus building tools looking forward is a move away from saving and exchanging data in local files on annotators' computers or private servers. Corpora are increasingly built using public, version-controlled repositories on platforms

such as GitHub. For example, the Universal Dependencies project is managed entirely on GitHub, including publically available data in multiple languages and the use of GitHub pages and issue trackers for annotation guidelines and discussion. Some tools (e.g. the online XML and spreadsheet editor GitDox, Zhang & Zeldes 2017) are opting for online storage on GitHub and similar platforms as their exclusive file repository. In the future we will hopefully see increasing openness and interoperability between tools which adopt open data models and best practices that allow users to benefit from and re-use existing data and software.

5. Resources

5.1 Tools

An important set of tools influencing the choice of corpus architecture is NLP pipelines and APIs, which allow users to construct automatically tagged and parsed representations with complex data models (and these can be manually corrected if needed). Some examples include Stanford CoreNLP (Manning et al. 2014), Apache OpenNLP (https://opennlp.apache.org/), Spacy (https://spacy.io/), the Natural Language Toolkit (NLTK, http://www.nltk.org/), GATE (Cunningham et al. 1997), DKPro (Eckart de Castilho & Gurevyich 2014), NLP4J (https://emorynlp.github.io/nlp4j/) and FreeLing (http://nlp.cs.upc.edu/freeling/).

The output formats of NLP tools is often not compatible with corpus search architectures, and may not be readily human-readable (for example, .json files offer very efficient storage, but are only meant to be machine readable). For this reason, NLP tool output must often be converted into corpus formats such as those in Table 1. Versatile conversion tools, such as Pepper (http://corpus-tools.org/pepper/), can be used to convert between a variety of formats and make data accessible to a wider range of tools. Another important feature supported by tools such as Pepper is merging data from several formats into a format capable of expressing the multiple streams of input data. Using a merging paradigm makes it possible to build corpora that require some advanced features (e.g. conflicting spans, or multimodal time alignment), which are not available simultaneously in the tools we wish to use, but can be represented separately in a range of tools, only to be merged later on. For example, the GUM corpus described above is annotated using five different tools which are optimized to specific tasks, and the merged representation is

created automatically (this is sometimes called a 'build bot' strategy; for an example see https://corpling.uis.georgetown.edu/gum/build.html).

Finally, corpus architecture considerations also interact with the choice of search and visualization facilities that one intends to use. Having an annotation tool which supports a complex data model may be of little use if the annotated data cannot be accessed and used in sensible ways later on. Some corpus practitioners use scripts, often in Python or R, to evaluate their data, without using a dedicated search engine (see Chapter 9, Programming for Corpus Linguistics). While this approach is very versatile, it is also labor intensive: for each new type of information, a new script must be written which traverses the corpus in search of some information. It is therefore often desirable to have a search engine that is capable of extracting data based on a simple query. For corpora that are meant to be publically available to non-expert users, this is a necessity. In public projects, a proprietary search engine tailored explicitly for a specific corpus is often programmed, which cannot easily be used for other corpora. Here I therefore focus on generic, freely available tools which can be used for a variety of datasets.

The Corpus Workbench (Christ 1994) and its web interface CQPWeb (Hardie 2012) are amongst the most popular tools for corpus search and visualization, but are not capable of representing hierarchical data, and therefore they cannot be used for treebanks. Grid-like data, e.g. from EXMARaLDA or Elan files, can be indexed for search using EXMARaLDA's search engine, EXAKT (http://exmaralda.org/en/exakt-en/). For treebanks, there are some local user tools (e.g. TigerSearch, Lezius 2002, or command line tools such as TGrep2, http://tedlab.mit.edu/~dr/Tgrep2/, the successor of the original Penn Treebank tool, or Stanford's Tregex, https://nlp.stanford.edu/software/tregex.shtml). There are only a few dedicated web interfaces for treebanks, notably Ghodke & Bird's (2012) highly efficient Fangorn (for projective, unlabeled constituent trees), and TüNDRA, the Tübingen aNnotated Data Retrieval Application, for TigerXML style trees and dependency trees (Martens 2013). For small-medium sized multilayer corpora, with syntax trees, entity and coreference annotation, discourse parses and more, ANNIS (http://corpus-tools.org/annis/) offers a comprehensive solution supporting highly complex graph queries over hierarchies, conflicting spans, aligned A/V data and parallel

corpora. For larger datasets, KorAP (Diewald et al. 2016) presents a search engine supporting a substantial subset of graph relations, accelerated for text search using Apache Lucene.

5.2 Further reading

For readers with some corpus building experience, Kübler & Zinsmeister (2015) gives a comprehensive overview of many aspects of complex annotated corpora, including data models and corpus query languages for treebanks and multilayer corpora. McEnery et al. (2006) is a good hands-on introduction for readers with less background, and presents and discusses both central readings on corpus design and practical case studies with a variety of corpora, including multilingual data and alignment. Weisser (2016) is also a practical guide for beginners, focusing on flat-annotated corpora, as well as working with some more complex data, such as the British National Corpus (BNC, http://www.natcorp.ox.ac.uk/). More information on specific topics in corpus architecture can also be found in selected chapters from Lüdeling & Kytö (2008-2009).

References

Bański, Piotr (2010). 'Why TEI Stand-off Annotation Doesn't Quite Work and Why You Might Want to Use it Nevertheless', in *Proceedings of Balisage: The Markup Conference 2010*. Montréal.

Biber, Douglas (1993). 'Representativeness in Corpus Design', *Literary and Linguistic Computing* 8(4): 243–257.

Boyd, Adriane, Hana, Jirka, Nicolas, Lionel, Meurers, Detmar, Wisniewski, Katrin, Abel, Andrea, Schöne, Karin, Štindlová, Barbora, and Vettori, Chiara (2014). The MERLIN Corpus: Learner Language and the CEFR. In: *Proceedings of LREC 2014*. Reykjavik, Iceland, 1281–1288.

Brugman, Hennie, and Russel, Albert (2004). 'Annotating Multimedia/Multi-modal resources with ELAN', in *Proceedings of LREC 2004*. Paris: ELRA, 2065–2068.

Burchardt, Aljoscha, Padó, Sebastian, Spohr, Dennis, Frank, Anette, and Heid, Ulrich (2008). 'Formalising Multi-layer Corpora in OWL DL - Lexicon Modelling, Querying and Consistency Control', in *Proceedings of IJCNLP 2008*. Hyderabad, India, 389–396.

Calzolari, Nicoletta, and McNaught, John (1994). *EAGLES Interim Report EAG--EB--IR--2*.

Canales, Omar, Monaco, Vinnie, Murphy, Thomas, Zych, Edyta, Stewart, John, Castro, Charles Tappert Alex, Sotoye, Ola, Torres, Linda, and Truley, Greg (2011). 'A Stylometry System for Authenticating Students Taking Online Tests', in *Proceedings*

of Student-Faculty Research Day, CSIS, Pace University, May 6th, 2011. White Plains NY, B4.1–B4.6.

Castilho, Richard Eckart de, and Gurevych, Iryna (2014). 'A Broad-Coverage Collection of Portable NLP Components for Building Shareable Analysis Pipelines', in *Proceedings of the Workshop on Open Infrastructures and Analysis Frameworks for HLT*. Dublin, 1–11.

Castilho, Richard Eckart de, Ide, Nancy, Lapponi, Emanuele, Oepen, Stephan, Suderman, Keith, Velldal, Erik, and Verhagen, Marc (2017). 'Representation and Interchange of Linguistic Annotation: An In-Depth, Side-by-Side Comparison of Three Designs', in *Proceedings of the 11th Linguistic Annotation Workshop (LAW XI)*. Valencia, Spain, 67–75.

Christ, Oliver (1994). 'A Modular and Flexible Architecture for an Integrated Corpus Query System', in *Proceedings of Complex 94. 3rd Conference on Computational Lexicography and Text Research*. Budapest, 23–32.

Crasborn, Onno, and Sloetjes, Han (2008). 'Enhanced ELAN Functionality for Sign Language Corpora', in *Proceedings of the 3rd Workshop on the Representation and Processing of Sign Languages at LREC 2008*. Marrakesh, Morocco, 39–42.

Cunningham, Hamish, Humphreys, Kevin, Gaizauskas, Robert, and Wilks, Yorick (1997). 'Software Infrastructure for Natural Language Processing', in *Proceedings of the Fifth Conference on Applied Natural Language Processing*. Washington, DC, 237–244.

Diewald, Nils, Hanl, Michael, Margaretha, Eliza, Bingel, Joachim, Kupietz, Marc, Bański, Piotr, and Witt, Andreas (2016). 'KorAP Architecture - Diving in the Deep Sea of Corpus Data', in *Proceedings of LREC 2016*. Portorož: ELRA.

Dipper, Stefanie (2005). 'XML-based Stand-off Representation and Exploitation of Multi-Level Linguistic Annotation', in *Proceedings of Berliner XML Tage 2005*. Berlin, Germany, 39–50.

Garside, Roder, Leech, Geoffrey, and Sampson, Geoffrey (eds.) (1987). *The Computational Analysis of English: A Corpus-based Approach*. London: Longman.

Ghodke, Sumukh, and Bird, Steven (2010). 'Fast Query for Large Treebanks', in *Proceedings of NAACL 2010*. Los Angeles, CA, 267–275.

Green, Lisa J. (2002). *African American English: A Linguistic Introduction*. Cambridge: Cambridge University Press.

Greenbaum, Sidney (ed.) (1996). *Comparing English Worldwide: The International Corpus of English*. Oxford: Clarendon Press.

Hardie, Andrew (2012). 'CQPweb - Combining Power, Flexibility and Usability in a Corpus Analysis Tool', *International Journal of Corpus Linguistics* 17(3): 380–409.

Hearst, Marti A. (1997). 'TextTiling: Segmenting Text into Multi-paragraph Subtopic Passages', *Computational Linguistics* 23(1): 33–64.

Heiden, Serge (2010). ‘The TXM Platform: Building Open-Source Textual Analysis Software Compatible with the TEI Encoding Scheme’, in *24th Pacific Asia Conference on Language, Information and Computation*. Sendai, Japan, 389–398.

Hinrichs, Erhard W., Hinrichs, Marie, and Zastrow, Thomas (2010). ‘WebLicht: Web-Based LRT Services for German’, in *Proceedings of the ACL 2010 System Demonstrations*. Uppsala, 25–29.

Höder, Steffen (2012). ‘Annotating Ambiguity: Insights from a Corpus-Based Study on Syntactic Change in Old Swedish’, in T. Schmidt, and K. Wörner (eds.), *Multilingual Corpora and Multilingual Corpus Analysis*. (Hamburg studies on multilingualism 14.) Amsterdam and Philadelphia: Benjamins, 245–271.

Hovy, Eduard, Marcus, Mitchell, Palmer, Martha, Ramshaw, Lance, and Weischedel, Ralph (2006). ‘OntoNotes: The 90% Solution’, in *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*. New York, 57–60.

ISO 24615 (2010). *Language Resource Management – Syntactic Annotation Framework (SynAF)*.

ISO 24612 (2012). *Language Resource Management – Linguistic Annotation Framework (LAF)*.

Ide, Nancy, Baker, Collin, Fellbaum, Christiane, and Passonneau, Rebecca (2010). ‘The Manually Annotated Sub-Corpus: A Community Resource for and by the People’, in *Proceedings of ACL 2010*. Uppsala, Sweden, 68–73.

Ide, Nancy, and Suderman, Keith (2007). ‘GrAF: A Graph-based Format for Linguistic Annotations’, in *Proceedings of the Linguistic Annotation Workshop 2007*. Prague, 1–8.

Klein, Thomas, and Dipper, Stefanie (2016). *Handbuch zum Referenzkorpus Mittelhochdeutsch*. (Bochumer Linguistische Arbeitsberichte 19.) Bochum: Universität Bochum Sprachwissenschaftliches Institut.

Kountz, Manuel, Heid, Ulrich, and Eckart, Kerstin (2008). ‘A LAF/GrAF-based Encoding Scheme for Underspecified Representations of Dependency Structures’, in *Proceedings of LREC 2008*. Marrakesh, Morocco.

Krause, Thomas, Lüdeling, Anke, Odebrecht, Carolin, and Zeldes, Amir (2012). ‘Multiple Tokenizations in a Diachronic Corpus’, in *Exploring Ancient Languages through Corpora*. Oslo.

Krause, Thomas, and Zeldes, Amir (2016). ‘ANNIS3: A New Architecture for Generic Corpus Query and Visualization’, *Digital Scholarship in the Humanities* 31(1): 118–139.

Kredens, Krzysztof, and Coulthard, Malcolm (2012). ‘Corpus Linguistics in Authorship Identification’, in P. M. Tiersma, and L. M. Solan (eds.), *The Oxford Handbook of Language and Law*. Oxford: Oxford University Press, 504–516.

Kübler, Sandra, and Zinsmeister, Heike (2015). *Corpus Linguistics and Linguistically Annotated Corpora*. London: Bloomsbury.

Kučera, Henry, and Francis, W. Nelson (1967). *Computational Analysis of Present-day English*. Providence: Brown University Press.

Kupietz, Marc, Belica, Cyril, Keibel Holger, and Witt, Andreas (2010). The German Reference Corpus DEREKO: A Primordial Sample for Linguistic Research. In: *Proceedings of LREC 2010*. Valletta, Malta, 1848–1854.

Lee, John, Yeung, Chak Yan, Zeldes, Amir, Reznicek, Marc, Lüdeling, Anke, and Webster, Jonathan (2015). 'CityU Corpus of Essay Drafts of English Language Learners: A Corpus of Textual Revision in Second Language Writing', *Language Resources and Evaluation* 49(3): 659–683.

Leech, Geoffrey N. (1997). 'Introducing Corpus Annotation', in R. Garside, G. N. Leech, and T. McEnery (eds.), *Corpus Annotation: Linguistic Information from Computer Text Corpora*. London and New York: Routledge, 1–18.

Lezius, Wolfgang (2002). *Ein Suchwerkzeug für syntaktisch annotierte Textkorpora*. PhD Thesis, Institut für maschinelle Sprachverarbeitung Stuttgart.

Lüdeling, Anke, and Kytö, Merja (eds.) (2008-2009). *Corpus Linguistics. An International Handbook*. (Handbooks of Linguistics and Communication Science 29.) Berlin and New York: Mouton de Gruyter.

Lüdeling, Anke, Walter, Maik, Kroymann, Emil, and Adolphs, Peter (2005). 'Multi-level Error Annotation in Learner Corpora', in *Proceedings of Corpus Linguistics 2005*. Birmingham, UK.

Lundborg, Joakim, Marek, Torsten, Mettler, Maël, and Volk, Martin (2007). 'Using the Stockholm TreeAligner', in *Proceedings of the Sixth Workshop on Treebanks and Linguistic Theories*. Bergen.

Mann, William C., and Thompson, Sandra A. (1988). 'Rhetorical Structure Theory: Toward a Functional Theory of Text Organization', *Text* 8(3): 243–281.

Manning, Christopher D., Surdeanu, Mihai, Bauer, John, Finkel, Jenny, Bethard, Steven J., and McClosky, Davide (2014). 'The Stanford CoreNLP Natural Language Processing Toolkit', in *Proceedings of ACL 2014: System Demonstrations*. Baltimore, MD, 55–60.

Marcus, Mitchell P., Santorini, Beatrice, and Marcinkiewicz, Mary Ann (1993). 'Building a Large Annotated Corpus of English: The Penn Treebank', *Special Issue on Using Large Corpora, Computational Linguistics* 19(2): 313–330.

Martens, Scott (2013). 'Tundra: A Web Application for Treebank Search and Visualization', in *Proceedings of the Twelfth Workshop on Treebanks and Linguistic Theories (TLT12)*. Sofia, 133–144.

McEnery, Tony, Xiao, Richard, and Tono, Yukio (2006). *Corpus-Based Language Studies: An Advanced Resource Book*. (Routledge Applied Linguistics.) London and New York: Routledge.

Nivre, Joakim (2008). ‘Treebanks’, in A. Lüdeling, and M. Kytö (eds.), *Corpus Linguistics. An International Handbook*. Vol. 1. Berlin: Mouton de Gruyter, 225–241.

Odebrecht, Carolin, Belz, Malte, Zeldes, Amir, and Lüdeling, Anke (2016). ‘RIDGES Herbology - Designing a Diachronic Multi-Layer Corpus’, *Language Resources and Evaluation*.

Reznicek, Marc, Lüdeling, Anke, Krummes, Cedric, Schwantuschke, Franziska, Walter, Maik, Schmidt, Karin, Hirschmann, Hagen, and Andreas, Torsten (2012). *Das Falko-Handbuch. Korpusaufbau und Annotationen*. Humboldt-Universität zu Berlin, Technical Report, Version 2.01, Berlin.

Reznicek, Marc, Lüdeling, Anke, and Hirschmann, Hagen (2013). ‘Competing Target Hypotheses in the Falko Corpus: A Flexible Multi-Layer Corpus Architecture’, in A. Díaz-Negrillo, N. Ballier, and P. Thompson (eds.), *Automatic Treatment and Analysis of Learner Corpus Data*. Amsterdam: John Benjamins, 101–124.

Romary, Laurent, and Bonhomme, Patrice (2000). ‘Parallel Alignment of Structured Documents’, in J. Véronis (ed.), *Parallel Text Processing: Alignment and Use of Translation Corpora*. Dordrecht: Kluwer, 201–217.

Romary, Laurent, Zeldes, Amir, and Zipser, Florian (2015). ‘<tiger2/> – Serialising the ISO SynAF Syntactic Object Model’, *Language Resources and Evaluation* 49(1): 1–18.

Santorini, Beatrice (1990). *Part-of-Speech Tagging Guidelines for the Penn Treebank Project (3rd Revision)*. Technical Report, University of Pennsylvania.

Sauer, Simon, and Lüdeling, Anke (2016). ‘Flexible Multi-Layer Spoken Dialogue Corpora’, *International Journal of Corpus Linguistics, Special Issue on Spoken Corpora* 21(3): 419–438.

Schembri, Adam, Fenlon, Jordan, Rentelis, Ramas, Reynolds, Sally, and Cormier, Kearsy (2013). ‘Building the British Sign Language Corpus’, *Language Documentation and Conservation* 7: 136–154.

Schmid, Helmut (1994). ‘Probabilistic Part-of-Speech Tagging Using Decision Trees’, in *Proceedings of the Conference on New Methods in Language Processing*. Manchester, UK, 44–49.

Schmid, Helmut (2008). ‘Tokenizing and Part-of-Speech Tagging’, in A. Lüdeling, and M. Kytö (eds.), *Corpus Linguistics. An International Handbook*. Vol. 1. Berlin: Mouton de Gruyter, 527–551.

Schmidt, Thomas, and Wörner, Kai (2009). ‘EXMARaLDA – Creating, Analysing and Sharing Spoken Language Corpora for Pragmatic Research’, *Pragmatics* 19(4): 565–582.

Smith, Jason R., Quirk, Chris, and Toutanova, Kristina (2010). ‘Extracting Parallel Sentences from Comparable Corpora using Document Level Alignment’, in *Proceedings of NAACL 2010*. Los Angeles, 403–411.

Tenfjord, Kari, Meurer, Paul, and Hofland, Knut (2006). The ASK Corpus - A Language Learner Corpus of Norwegian as a Second Language. In: *Proceedings of LREC 2006*. Genoa, Italy, 1821–1824.

Weisser, Martin (2016). *Practical Corpus Linguistics: An Introduction to Corpus-Based Language Analysis*. Oxford: Wiley Blackwell.

Wichmann, Anne (2008). Speech Corpora and Spoken Corpora. In: Anke Lüdeling & Merja Kytö (eds.), *Corpus Linguistics. An International Handbook*. Vol. 1. Berlin: Mouton de Gruyter, 187–207.

Yimam, Seid Muhie, Gurevych, Iryna, Castilho, Richard Eckart de, and Biemann, Chris (2013). 'WebAnno: A Flexible, Web-based and Visually Supported System for Distributed Annotations', in *Proceedings of ACL 2013*. Sofia, Bulgaria, 1–6.

Zeldes, Amir (2017). 'The GUM Corpus: Creating Multilayer Resources in the Classroom', *Language Resources and Evaluation* 51(3), 581-612.

Zeldes, Amir (forthcoming). *Multilayer Corpus Studies*. (Routledge Advances in Corpus Linguistics.) London: Routledge.

Zhang, Shuo & Amir Zeldes (2017). GitDOX: A Linked Version Controlled Online XML Editor for Manuscript Transcription. In: *Proceedings of FLAIRS-30*. Marco Island, FL, 619–623.

Zipser, Florian, and Romary, Laurent (2010). 'A Model Oriented Approach to the Mapping of Annotation Formats using Standards', in *Proceedings of the Workshop on Language Resource and Language Technology Standards, LREC-2010*. Valletta, Malta, 7–18.