1

Annotating Collaboratively

1.1. The annotation process (re)visited

A simplified representation of the annotation process is shown in Figure 1.4. We will detail further in this section the different steps of this process, but we first introduce a theoretical view on the consensus and show how limited the state of the art on the subject is.

1.1.1. *Building consensus*

The central question when dealing with manual corpus annotation is how to obtain reliable annotations, that are both useful (i.e. meaningful) and consistent. In order to achieve this and to solve the "annotation conundrum" [LIB 09], we have to understand the annotation process. As we saw in section I.1.2, annotating consists of identifying the segment(s) to annotate and adding a note (also called a label or a tag) to it or them. In some annotation tasks, segments can be linked by a relation, oriented or not, and the note applies to this relation. In most cases, the note is in fact a category, taken from a list (the tagset).

Alain Desrosières, a famous French statistician, worked on the building of the French socio-professional categories

[DES 02] and wrote a number of books on categorization (among which, translated into English, [DES 98]). His work is especially relevant to our subject, as he precisely analyzed what categorizing means.

First, and this is fundamental for the annotation process, he makes a clear distinction between *measuring* and *quantifying* [DES 14]. Measuring "implies that something already exists under a form that is measurable, according to a realistic metrology, like the height of the Mont Blanc".[1] Quantifying, on the other hand, consists of "expressing and transforming into a numerical form what used to be expressed with words and not numbers".[2] For this to be realized, a series of conventions of equivalence should be elaborated through collaboration.

The categories are not measurable, they have to be agreed upon before they can be applied. There has to be a consensus on them and one piece of evidence that the categories emerge from a consensus (and are not "natural") is that they can change in time. A typical example of this are named entities, which evolved from proper names only [COA 92] to the MUC (Message Understanding Conferences) classic categories (*person, location, organization*) [GRI 96] and on to structured named entities, with subtypes and components [GRO 11]. This evolution was initiated and validated by the named entity recognition community. This also happened, although in a less spectacular way, with parts-of-speech [COL 88].

The result of this consensus-building process is logged in the annotation guidelines, that are used by the annotators to decide what to annotate (which segment(s)) and how (with

1 In French: "[...] l'idée de mesure [...] implique que quelque chose existe *déjà* sous une forme mesurable selon une métrologie réaliste, comme la hauteur du Mont Blanc".

2 In French: "exprimer et faire exister sous une forme numérique ce qui auparavant, était exprimé par des mots et non par des nombres."

which category). However, even with very detailed guidelines, like the 80 pages long *Quæro* structured named entity annotation guidelines,[3] the annotators will still disagree on some annotations. This is why we need constant evaluation (to see when they disagree) and consensus building (to improve the consistency of annotations).

Once this is posited, there remain many practical issues: who should participate in the annotation guidelines? and how can we determine when they are ready, or at least ready-enough to start annotating? When do we start evaluating the agreement between annotators, and how? The following sections will hopefully provide answers to these questions.[4]

1.1.2. *Existing methodologies*

Manual annotation has long been considered as straightforward in linguistics and NLP. Some researchers still consider that computing inter-annotator agreement is useless (since the annotators *have to* agree) and it took some time and demonstration [NÉD 06] before the need for an annotation guide became obvious. It is therefore logical that the interest for the manual annotation process itself is growing slowly.

If speech processing inspired the evaluation trend and metrics like inter-annotator agreements, corpus linguistics provided good practices for manual annotation, in particular with Geoffrey Leech's seven maxims [LEE 93] and later work on annotation [LEE 97], and with collective efforts like [WYN 05]. However, it did not propose any in-depth analysis of the annotation process itself.

---

3 Available here: http://www.quaero.org/media/files/bibliographie/quaero-guide-annotation-2011.pdf.

4 The sections about the annotation process, from preparation to finalization, are adapted from my PhD thesis (in French) [FOR 12a].

Some high-level analyses of the work of the annotators were carried out, for example to create the *UniProt* Standard Operating Procedures[5] or the GATE manual.[6] However, very few studies are concerned with the manual annotation process as a whole.

According to Geoffrey Sampson [SAM 00], the "problem analysis and documentation" of annotation should be taken much more seriously and be considered primary over coding (annotating). His reflection is based on a parallel with software development and engineering. Interestingly, this parallel has been extended to the annotation methodology with "agile corpus creation" and "agile annotation" [VOO 08], an analogy with agile development [BEC 11].

From our point of view, the methodology presented in [BON 05], even if it is generally not cited as a reference for agile annotation, pioneered the field. The authors show that computing inter-annotator agreement very early in the campaign allows them to identify problems rapidly and to update the annotation guide accordingly, in order to minimize their impact.

Agile annotation [VOO 08] goes further as it reorganizes completely the traditional phases of manual annotation (see Figure 1.1) for a more lenient process, with several cycles of annotation/guideline update. To our knowledge, this methodology was used only once in a real annotation project [ALE 10]. Therefore, it is difficult to understand to what extent it really differs from the methodology presented in [BON 05] and whether it will produce better results.

5 See: http://www.uniprot.org/help/manual_curation and http://geneontology.org/page/go-annotation-standard-operating-procedures.

6 See: https://gate.ac.uk/teamware/man-ann-intro.pdf.

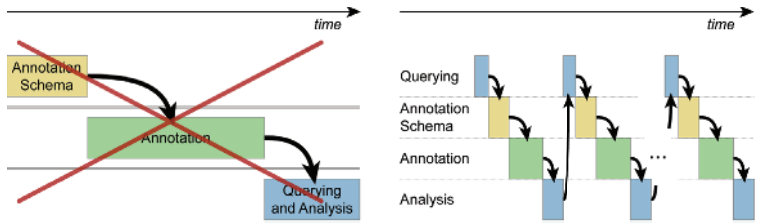


Figure 1.1. *Traditional annotation phases (on the left) and cycles of agile annotation (on the right). Reproduction of Figure 2 from [VOO 08], by courtesy of the authors*

Eduard Hovy presented a tutorial on manual annotation during the ACL 2010 conference in which he gave interesting insights about methodology and process. This partial methodology is detailed in [HOV 10] and shown in Figure 1.2. It includes the training and evaluation of the system (*engine*), the results of which can lead to modification of the manual annotation. Our point of view on this is quite different and we have already expressed it in section I.2.2: manual annotation should be carried out with an application in mind, not in accordance with a tool, as (i) it would largely bias any evaluation performed with the annotated corpus and (ii) would limit the lifespan of the corpus. However, the manual annotation part of this methodology is the most complete we know of. It includes six steps: (1) building the corpus, (2) developing the tagset and writing a first version of the guidelines, (3) annotating a sample of the corpus, (4) comparing the annotators' decisions, (5) measuring the inter-annotator agreement and determining which level of agreement would be satisfactory (if not, return to step 2), (6) annotating the corpus. Although it includes a pre-campaign (steps 2 to 5), post-campaign (delivery and maintenance) and consensus building elements (meetings), it neither defines who does what (the precise roles), nor gives indicators in order to move up one step, particularly concerning the training of the annotators.

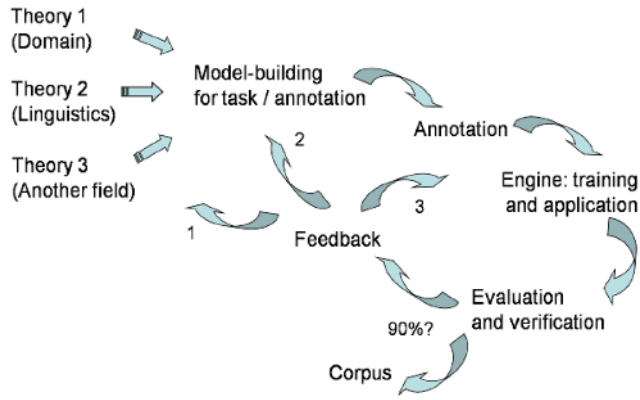


Figure 1.2. *Generic annotation pipeline (Figure 1 from [HOV 10], by courtesy of the authors)*

The book written by James Pustejovsky and Amber Stubbs [PUS 12] also presents a view on annotation where the training of systems and the manual process interpenetrate. This is the MATTER methodology (for *Model-Annotate-Train-Test-Evaluate-Revise*). Within MATTER lies the manual annotation cycle itself, MAMA (for *Model-Annotate-Model-Annotate*). In this, annotation is further decomposed into another cycle: Model and Guidelines-Annotate-Evaluate-Revise. According to MAMA, the corpus is entirely annotated by at least two annotators, several times, then completely adjudicated by an expert. This is ideal, but very costly if not done with crowdsourcing. Another weak point in this methodology is that it contains various cycles and does not indicate when they should be stopped.

We will focus here on the annotation process and will not detail the corpus creation, although it is a very important step. Useful information on the subject can be found in [PUS 12], but we advise reading John Sinclair first, in particular the easily accessible [SIN 05].

We have participated in a dozen annotation campaigns (in half of them, as campaign manager), most of them within the framework of the *Quæro* project. These campaigns cover a wide range of annotation types (and were carried out either in French or in English): POS, dependency syntax, named entities, gene renaming relations, protein and gene names, football, pharmacology. They allowed us to build and test the annotation process framework we propose in the following sections.

1.1.3. *Preparatory work*

An annotation campaign does not start with the annotation itself. It requires some preparatory work to identify the actors, to get to know the corpus and to write a first version of the guidelines. This should not be neglected, as the productivity and quality of the annotation largely depend on it.

1.1.3.1. *Identifying the actors*

The following reflection was started with Sophie Rosset (LIMSI-CNRS). It aims at identifying clearly the various actors in an annotation campaign and showing the tensions that can emerge from their often diverging visions of the campaign.

Our experience and the state-of-the-art allow us to distinguish between seven main roles in an annotation campaign:

1) final users: users of the potential application developed using the annotated corpus;

2) financier(s): person(s) representing the organism(s) funding the campaign (often funding agencies);

3) client(s): person(s) or team(s) who need the corpus to train, create or evaluate their system;

4) campaign manager: person in charge of planning the campaign and guaranteeing its performance. In general,

the manager is the contact person between the client, the evaluators and the experts (and in some rarer cases the financiers);

5) expert annotators: annotators who are specialized in the domain of the corpus (sometimes in the annotation domain), who select the annotators, train them, evaluate them, answer their questions and adjudicate the annotation when necessary;

6) annotators: persons performing the biggest part of the annotation; in crowdsourcing annotation they are sometimes called "non-experts", but we will see in the next chapter that this is far from the reality;

7) evaluator(s): person(s) in charge of evaluating the quality of the annotated corpus and/or of the systems trained or evaluated with this corpus.

All these roles are not always fulfilled in an annotation campaign. For example, evaluators can be absent from smaller internal campaigns, with no external evaluation. As for the number of actors per role, it can vary, but the annotation manager should be a unique person, in order to avoid inconsistencies in the campaign. As for experts, there should be at least two to build a mini-reference, so if the manager is an expert, there has to be another one. Finally, we will see in section 1.4 that to evaluate the quality of the annotation, at least two annotators are needed. In the meta study presented in [BAY 11], the authors concluded by the suggestion to use at least five annotators for the most difficult tasks, and at least three or four annotators for other tasks. Previously, it was shown in [KLE 09] that the use of more annotators enabled us to lower the influence of chance on the inter-annotator agreement results. However, it was demonstrated in [BHA 10] that well-trained annotators produce better annotations than a "crowd of non-experts". As we will see further, the number of annotators is not the key, but training is.

Obviously, all the roles we presented here should not necessarily be perfectly distinct and organized as a hierarchy in all campaigns, but when they are missing or merged with another role, it should be taken into account, since it may generate biases. For example, the financier, being in the most powerful position, could bias the whole campaign. Therefore, he/she should not intervene directly on the annotators or the annotation guide. Futhermore, when the financier is not the client, the balance of the campaign is more sound. With no financier, the client has a lot of influence (maybe too much). As for the manager, he/she is accountable for the overall balance of the campaign and should not play any other role. If he/she is also an expert (which is often the case), he/she has to work with other experts to compensate this imbalance. Finally, the expert has not only to supervise the annotators, but also to be their representative. It should be noted that, although the annotator is at the bottom of this organization, he/she is at the center of the annotation, as the value added to the corpus is the interpretation provided by the annotators. It is therefore essential to take their remarks and suggestions into account. Annotators on microworking platforms like `Amazon Mechanical Turk` are paid by the task and have no interest in giving feedback to the *Requester* (and the platform does not encourage them to do so), their point of view is therefore seldom considered.

Finally, we would like here to nuance a common statement according to which the researchers, who are often both managers of the campaign and experts of the annotation task, are the best annotators. Our experience shows that (i) even if they are experts of the task, they are not necessarily experts of the domain and can experience difficulties understanding the context, like in the case of named entity annotation in old press (what was “Macé” in “krach Macé”? A person? An organization? A place?) and (ii) they too often question the annotation guide they wrote or they do not consult it enough. During the structured named

entity annotation campaign in broadcast news, the four experts (researchers) who annotated the mini-reference obtained inter-agreement scores which were not better than that of the annotators.

1.1.3.2. *Taking the corpus into account*

We managed a football annotation campaign in which the heterogeneity of the corpus affected all the aspects of the campaign: the selection of a sub-corpus for the training of the annotators, the length of the training, the complexity of the annotation scheme and the resulting annotation quality. Based on this experience, we showed in [FOR 11b] how important it is to have an in-depth knowledge of the corpus to annotate.

This is all the more true as the campaign manager does not necessarily choose the corpus on which the annotation will be performed, he/she therefore has to adapt the campaign to the specificities of the source. This means that the corpus should be analyzed and decomposed into its constituents: domains, sources, media, etc.

The best way to "dive into" the corpus is to annotate a small but representative part of it, even before starting the campaign. Obviously, this is possible only if the domain and the language are mastered by the manager. If not, he/she should use one or several experts to help with this work.

This is what we did in several annotation campaigns, as campaign manager [FOR 11b] or as advisor [ROS 12]. It allowed us not only to identify problems with the annotation guide even before the annotators started working, but also to create a pre-reference for evaluation. In some cases, having the client annotate this pre-reference is a good way to validate the choices that were made and to check that the annotation is not diverging too much from the initial application.

Whether it is done *a priori*, during the corpus selection, or *a posteriori*, once the corpus is selected, a precise analysis of its contents and of the consequences of this on the campaign has to be performed as soon as possible.

1.1.3.3. *Creating and modifying the annotation guide*

The annotation guide (also called annotation guidelines) is now recognized as essential to an annotation campaign. For the structured named entity annotation campaign, the design of the annotation guide took six months. This preparatory work was costly (especially as it involved several researchers), even if the resulting guide has been used in a second annotation campaign, as well as for another French project (ETAPE).

However, writing an annotation guide is not a one-shot task performed at the beginning of a campaign, with only a couple of modifications added afterwards. On the contrary, the guide evolves during a large part of the annotation campaign. It is the necessary condition for its usability as the accompanying documentation for the resulting annotated corpus.

However, a first version of the guide should be written rapidly, before the campaign starts, in collaboration with the client (we call this a pre-reference). It is then tested by annotating a mini-reference. Usually, this generates a first round of modifications. During the break-in phase, the document will continue to be improved, thanks to the feedback from the annotators. In turn, these modifications should allow for a better quality of the annotation and for a gain in time, since the ill-defined or ill-understood categories and rules generate a waste of time for the annotators. Several cycles annotation/revision of the guide can be necessary to obtain a certain stability, which is demonstrated through a constant and sufficient annotation quality.

If their underlying principles are very close, agile annotation [VOO 08, ALE 10] differs from the methodology

proposed in [BON 05] in that the cycles continue until the very end of the campaign (see Figure 1.1). However, it seems to us that when the annotation is stabilized in terms of annotation quality and speed, it is not necessary to go on with the process, even if other evaluations should be performed to ensure non-regression.

Finally, ill-defined or ill-understood categories are a cause of stress and mistakes. In order to alleviate the stress and to keep a precise trace of the problems encountered during annotation, it is important to offer the annotators the possibility to add an uncertainty note when they have doubts about their decisions. This uncertainty note can take the form of typed features (for example, see Figure 1.10: uncertainty-type="too generic"), which allow for an easier processing. These types of uncertainties should of course be described in the annotation guide.

We give a number of recommendations concerning the annotation guide in [FOR 09]. We briefly summarize them here:

– indicate *what* should be annotated rather than *how*;

– do not *a priori* exclude what would be doubtful or too difficult to reproduce with a NLP system;

– give the annotators a clear vision of the application in view;

– add precise definitions, justify the methodological choices and explain the underlying logics of the annotation (do not just provide examples).

Following these recommendations should empower and motivate the annotators, by giving them access to the underlying logics. This way, we allow them to evolve from a "father-son" relationship to a pair relationship [AKR 91], which influences the annotation quality and is all the more necessary if the annotators are (corpus) domain experts who have to be as autonomous as possible.

It is therefore essential not to describe everything and to leave a sufficient interpretation margin to the annotators so that they can really add value to the corpus. Guidelines which are too detailed and long to consult are less useful than a condensed guide, presenting what is essential, with a few well-chosen examples and concrete tests to distinguish between the categories which are known to be ambiguous. From this point of view, the *Penn Treebank* guidelines for POS annotation are an example to follow.

In crowdsourcing, this principle is pushed to its maximum, as the annotation guide is reduced to a couple of lines on `Amazon Mechanical Turk`, or to a couple of pages for a gamified interface like `Phrase Detectives`. In these cases, the annotation task should remain simple or the training should replace at least part of the guidelines.

The preparatory work allows us to clearly define the application in view, to write a first version of the annotation guide, to explore the corpus and to identify the actors of the campaign. It includes three main phases: (i) the pre-campaign, during which a mini-reference is agreed upon and the annotators are trained, (ii) the annotation itself, which starts with a break-in period and includes regular evaluations and updates, and (iii) finalization, which consists of a manual or automatized correction of the annotated corpus, before its publication. The general organization of an annotation campaign is shown in Figure 1.4.

1.1.4. *Pre-campaign*

The consensus building phase is too often reduced to a couple of meetings, when it should be an iterative process that involves various actors. If the pre-campaign is organized by the campaign manager, he/she is generally associated with (annotation) domain experts in building the corpus sample which will be annotated to be used as a mini-reference.

He/she is also in charge of the training of the annotators, during which they will give the first feedback on the campaign (organization, tools, guidelines).

1.1.4.1. *Building the mini-reference*

Building a mini-reference from the very beginning of the campaign (see Figure 1.4) presents numerous advantages. First, it allows us to test in real conditions the first version of the annotation guide, written by the manager, sometimes in collaboration with the client. Building the mini-reference also allows us to evaluate the reliability of the annotation very early in the campaign. The result of this evaluation will be compared to others, later in the campaign. Moreover, once it is finalized, the mini-reference contains all the information needed to compute the complexity dimensions of the campaign (see section 1.2), that will give precise indications to select the most appropriate tools for the campaign, be they annotation tools (see section 1.3), pre-annotation tools or methodological solutions (for example adding elements to the guidelines). This step also allows us to select the most appropriate inter-annotator agreement metric (see section 1.4).

The reference sub-corpus (or mini-reference) is a sample from the original "raw" corpus, if possible representative. The preparatory work (see section 1.1.3) allowed us to establish a detailed typology of the corpus and the creation of a representative sub-corpus for the mini-reference can be done by selecting files (or parts of files) corresponding to each identified type, in a proportionate way. Our goal here is not to be perfectly representative (which is an illusion anyway), but to cover enough phenomena to deal with a maximum of issues during the annotation of the mini-reference.

The size of this sub-corpus mostly depends on the time available for this annotation, but a corpus that is too small or an insufficient representativeness can lead to important

errors in the computation of the complexity dimensions of the campaign. For example, we noticed when we computed the complexity dimensions for the structured named entity annotation campaign, that the selected sample was too small. The theoretical ambiguity is relatively limited on the mini-reference (around 0.15) and much higher on the global corpus (around 0.4). These results are detailed in [FOR 12d].

This mini-reference is annotated by the campaign manager (or by an expert, if the domain of the corpus is unknown to the manager), with at least one expert. The annotation phase is punctuated by informal meetings during which modifications of the tagset and of the guidelines are decided upon. Collective solutions are found to disagreements by consensus. We created mini-references for two annotation campaigns (football and structured named entities) and in both cases they were finalized late in the campaign, but were used for the evaluation.

In crowdsourcing annotation, such mini-references are quite common, and are used to validate the work of the participants. For example, in `Phrase Detectives` [CHA 08] and `ZombiLingo` [FOR 14b], a reference corpus annotated by experts of the task is used for the training and evaluation of the players.

It has to be noted that building a mini-reference represents a "mini-campaign" inside the campaign. Consequently, the steps described in sections 1.1.5 and 1.1.6 also apply to the mini-reference. However, in practice, the break-in period and the publication are not needed.

1.1.4.2. *Training the annotators*

The training of the annotators is now recognized as essential to the quality of the annotation (see, among others [DAN 09, BAY 11]) and should be taken into account in the annotation campaign.

Usually, the annotators are trained for the task, i.e. both on the annotation itself and on the tool used for it. However, the two trainings present different types of difficulties. For annotators who are very competent in their domain but not at ease with computers, it is important to find the most appropriate tool, even if it means being a little less efficient (for example, a point-and-click tool like `Glozz`). Note that getting familiar with the tool can take more time than expected for these annotators. The training phase can also be used to detect annotators who are unable to perform the task correctly and to exclude them.

The training is done on an extract from the mini-reference, which has to be annotated by the annotators using the annotation tool and according to the provided guidelines. If possible, a first collective training session, with all the annotators, is more profitable than distant training, as they can ask all the questions they want and get all the answers at once.

This first collective phase should be followed by another phase during which the annotators work in real conditions and in parallel, without consulting each other, on the same sub-corpus, tracking their time. This tracked time will be used to visualize the learning curve of the annotators, like we did with ours on the *Penn Treebank* (see Figure 1.3). This curve is the first indicator of the level of training of the annotators. The second indicator is the produced quality.

The evaluation of the training can be done on the mini-reference (accuracy or F-measure) or between annotators (inter-annotator agreement). A discussion should be organized with the annotators to explain the difficult points (the ones on which they disagree the most between themselves or with the reference).

The training phase can expose errors or imprecisions in the annotation guide and thus lead to modifications of the guidelines and of the mini-reference.

In games like Phrase Detectives or ZombiLingo, the training phase is automatized (indications are provided to the players to help them train themselves during the tutorial phase) and ends only when the player performs sufficiently well (less than 50% errors on Phrase Detectives for example).

On the contrary, in microworking platforms, the annotators can at best be submitted to a competency test before starting to work, but to our knowledge, no training phase is planned in the system.

We will see in the next chapter that training and crowdsourcing are not contradictory, but to associate them questions what some consider to be one of the fundamental principles of the system: the participation of "non-experts". Is training "non-experts" not the same as transforming them into experts, at least of the task?

1.1.5. *Annotation*

1.1.5.1. *Breaking-in*

The end of the pre-campaign does not immediately correspond to a definitive stabilization of the campaign. First, the training of the annotators will continue, since they rarely reach the maximum of their possibilities at the end of the pre-campaign (for the POS annotation of the *Penn Treebank*, the learning period lasted one month). Second, the annotation guide will be modified again, according to the annotators' remarks. Therefore, their annotations will possibly have to be corrected.

A more or less long break-in period thus succeeds to the pre-campaign. Depending on the available means, the

manager will continue to modify the guide more or less late in the campaign. The ideal would be to be able to review it until the very end of the campaign, in order to take into account all the elements discovered in the corpus. In practice, the guide needs to be stabilized so that the annotators can progress in the annotation and do not spend too much time correcting what they have already annotated. A good moment for that is probably when they reach their cruising speed (it can be detected easily as can be seen in Figure 1.3).

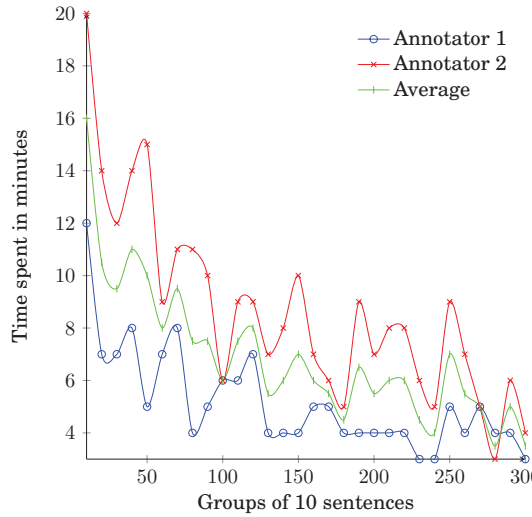


Figure 1.3. *Learning curve for the POS annotation of the* Penn Treebank *[FOR 10]. For a color version of the figure, see www.iste.co.uk/fort/nlp.zip*

This break-in phase also exists in crowdsourcing. The design of the game or of the microworking task requires several trials before the instructions (the minimal annotation guide), the interfaces (annotation tools) and the conditions of the annotation (for example with or without time limitation) are optimized. An example of these iterations is presented in [HON 11].

1.1.5.2. *Annotating*

The vast majority of the work will be carried out by the annotators during the annotation phase. The preceding steps allowed us to prepare it, but it is still important that the annotators be monitored by the expert or the manager on a regular basis.

Inter-annotator agreement metrics should be computed regularly, to check that the annotation is reliable (see Figure 1.4). This implies that at least partial parallelization is planned. Depending on the available time, the annotation can be performed totally in parallel, by at least two annotators, but most of the time only parts of it will be. In crowdsourcing, however, it is quite common to have the participants annotate all the corpus in parallel.

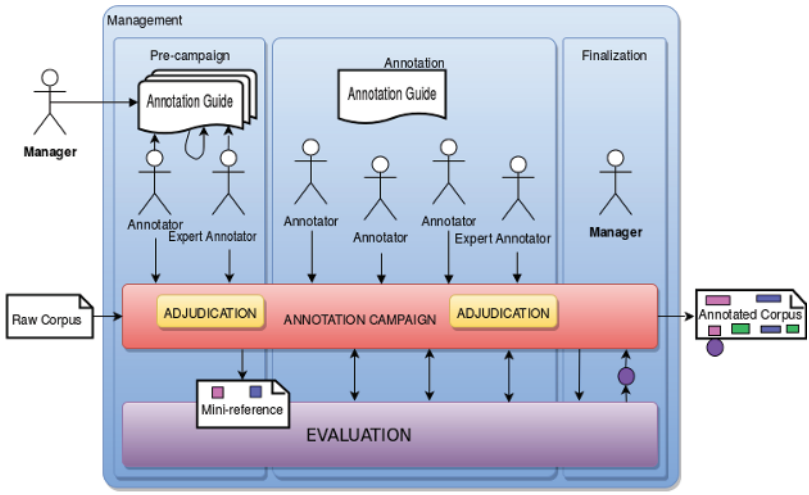


Figure 1.4. *The annotation process, revisited (simplified representation)*

The annotation phase itself can include an automatic pre-annotation step. In this case, the work of the annotators is limited to correcting this existing annotation and to

complete it if necessary. We carried on a very systematic study on pre-annotation with Benoît Sagot in [FOR 10], which showed that, at least on English POS annotation, there is a bias due to the pre-annotation (attention slips from the annotators, who rely too much on the pre-annotation). However, the observed gains are such (twice the speed of annotation, even with a low accuracy tool) that it is worth warning the annotators of the dangers in the guidelines. The task itself is not fundamentally different, so we will use the term annotation in those cases too.

Each annotator in the campaign is assigned some files to annotate. They are provided with the up-to-date annotation guide, which is coherent with the used data model, and an appropriate annotation tool. They have been trained for the task and they have assimilated the principles explained in the guide. The break-in period should have helped them in refining their understanding of the task.

Ideally, the guidelines should be directly integrated in the annotation tool and the tool should be able to check the conformity of the annotation with regard to the guidelines, but if this was possible, human annotators would no longer be needed. However, intermediary features exist, which allow for a more efficient usage of the guidelines. The first one consists of providing an easy access to the guidelines, using for example a hypertext link from the tool. Another one would be to have the tool apply constraints which are defined in the guidelines (this was done with `EasyRef` in the EASy campaign, see Appendix A.2.2). The minimum is to ensure that the guidelines and the data model used in the tool are consistent (`Slate` includes for example a general versioning of the annotation, see Appendix A.4.1).

The annotation tool used by the annotators should help them not only in annotating, but also in monitoring their progression on the files which were assigned to them, in tracking the time spent on each file (or on each annotation

level) and in notifying the expert or the manager of problems. It should also provide some advanced searching features (in the categories and in the text), so that the annotators can efficiently correct their annotations.

During the annotation phase, a regular evaluation of the conformity of the annotation with regards to the mini-reference should be done, associated with regular intra- and inter-annotator agreement measurements.

1.1.5.3. *Updating*

Even if it has been decided to stabilize the guidelines at the end of the break-in phase, updates are inevitable during the pre-campaign and the break-in phase. These updates have to be passed on to the annotated corpus, in order for it to remain consistent with the guidelines.

During the pre-campaign, updates are decided informally, between experts. The mini-reference being small by definition, the corrections can be made immediately.

During the break-in period, updates are either formally decided upon by the manager, following disappointing evaluations or less formally by the annotators, who ask the expert(s) or the manager to modify the guidelines. The manager can decide to give up on some of them for reasons of cost.

1.1.6. *Finalization*

Once the corpus is annotated, the manager has to finalize the campaign. He/she has at his/her disposal the annotations added to the corpus and a series of indicators, including at least evaluation metrics (conformity and intra- and inter-annotator agreement results), and sometimes uncertainty features added by the annotators. The manager can run a quick questionnaire among the annotators to try and catch

their impressions concerning the campaign. Then he/she has to decide what to do next. Four options are available:

1) publish the corpus, which is considered to be in a sufficiently satisfactory state to be final;

2) review the corpus and adapt the annotation guide;

3) adjudicate the corpus;

4) give up on revision and publication (failure).

In most cases, a correction phase is necessary. If the annotation was carried out totally in parallel by at least two annotators, this correction can correspond to an adjudication by an expert, but it is most of the time performed more or less automatically, using the indicators provided during the campaign.[7]

In case there is a correction (adjudication and reviewing), the corpus has to be evaluated and be submitted, with its indicators, to the decision of the manager, who can either publish the corpus or have it corrected again.

1.1.6.1. *Failure*

A complete failure, which would be noticed at the end of the campaign (during finalization) is a sign of an absence of management and remains rare. However, we witnessed such a case of failure in the campaign described in [FOR 09]. It was due to a series of causes, the main one being that there was no real manager in charge of the campaign.

1.1.6.2. *Adjudication*

The adjudication is the correction by one or more expert(s) of the annotations added by the annotators. This correction is usually limited to the disagreements between annotators

7 It has to be noted that corrections in the annotated corpus can also be done during the annotation phase itself. Therefore, it is important that the annotation tool provide a powerful search tool.

(hence the name), but we extend here its definition to the correction by an expert of all the annotations (a rare case in traditional annotation). In the first case, the expert validates (or not) one of the concurrent annotations. The annotations have therefore to be sorted prior to the adjudication, so that the expert only decides on disagreements. The expert can also be called for punctually, to decide on a case that is particularly difficult to annotate.

In all cases, the work of the expert can be facilitated using a tool, for example an adapted interface showing in parallel the conflicting annotations.

Interestingly, microworking *à la* `Amazon Mechanical Turk` does not exempt from manual correction. For example, in [KAI 08], PhD students were hired to validate the questions/answers corpus. In `Phrase Detectives` the corrections are made by the players themselves, who judge annotations added by others.

1.1.6.3. *Reviewing*

In most annotation campaigns, the available resources are not sufficient to manually correct the entire annotated corpus. The correction is therefore more or less automatized, from the indicators gathered during the annotation phase. When errors are consistent, they can be corrected globally on the whole corpus, without the need for an expert.

The manager (associated with an expert, if he/she is not one) can decide to merge two categories that are too ambiguous. The annotation then needs to be modified. He/she can also remove one or two categories if their annotation was problematic. Finally, he/she can decide not to take into account the annotations from a specific annotator, if they diverge too much (in particular in crowdsourcing).

Semi-automatic correction procedures were used in the structured named entity campaign in old press. These

corrections were identified thanks to a manual analysis of the errors carried on on a sample of the annotated corpus.

1.1.6.4. *Publication*

It is essential that the quality of the reviewed annotated corpus (or final corpus) is evaluated. In the case of a correction through adjudication of the disagreements, an evaluation performed by an expert of a random sample of uncorrected elements can be sufficient to evaluate the quality of the final corpus. In the (rare) case of a total correction of the annotated corpus, such a final evaluation is not needed, but can be carried out by a different expert on a sample of the corpus.

This final evaluation can be used as a seal of approval of the annotated corpus and can be taken into account during the evaluation of systems trained with this corpus. The corpus is published with its up-to-date annotation guide, if possible with a version number.

In all cases, the indicators provided with the annotated corpus are crucial to the manager.

1.2. Annotation complexity

What is complex? What should we automatize? In which case? An important step in the preparation of an annotation campaign is to identify the complexity dimensions of the annotation task at hand, as it allows us to better plan the annotation work and to put the right tools at the right place. However, this is far from being trivial, as everything seems entangled like in a wood ball.

We worked on the subject with Adeline Nazarenko (LIPN/University of Paris 13) and Sophie Rosset (LIMSI-CNRS), using the various annotation projects in which we

participated as a basis for our analysis. We identified and tested six complexity dimensions which we believe to be universal to all annotation tasks and we presented them in [FOR 12d]. We provide here what we hope to be a more pedagogical (and a slightly simplified) view on these complexity dimensions, trying to improve their presentation by taking into account the feedback we got on the main article.

The six complexity dimensions we will describe here are all independent of each other, except for one, the context. Identifying them for a specific annotation campaign means disentangling the wood ball. It may seem a little confusing at first, because we are not used to considering the complexity of a task as independent dimensions. However, this mental effort is essential to the deep understanding of an annotation task and we observed that the complexity grid we propose represents a very useful guide to changing perspectives on a campaign.

In order to be able to visualize the result globally without reforming another wood ball, we decided to associate metrics with each dimension, which, once computed, give results between 0 (null complexity) and 1 (maximum complexity). Some of these metrics can be computed *a priori* (without any annotation done yet) while others require an annotation sample or annotations from a similar campaign. Note that these metrics are independent from the volume to annotate and the number of annotators.

An example of visualization is shown in Figure 1.5 using a spiderweb diagram. The blue lines correspond to the dimensions linked to the identification of the segment to annotate, the three red lines to the dimensions related to the added note and the green one is the context. Instantiated examples will be given later on.

1.2.1. *Example overview*

First, let us have a look at examples of annotation in NLP. We take three, in which we participated either as annotators

(the *Penn Treebank* part-of-speech annotation,[8] and the structured named entity annotation) or as campaign manager (the gene renaming campaign). We believe they correspond to a large enough variety of situations to illustrate the complexity dimensions presentation we are going to make.

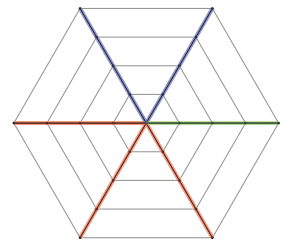


Figure 1.5. *Visualization of the complexity dimensions of an annotation task. For a color version of the figure, see www.iste.co.uk/fort/nlp.zip*

1.2.1.1. *Example 1: POS*

In the *Penn Treebank* part-of-speech (POS) annotation campaign, the corpus was pre-annotated and the annotators had to correct the provided annotations. As can be seen on Figure 1.6, the annotations were added in-line (inserted in the text itself),[9] separated from the original text by a simple marker (a slash), in a simple text editor. Like in any POS annotation campaign, all the lexical units[10] were annotated.

I/PRP do/VBP n't/RB feel/VB very/RB ferocious/JJ ./.

Figure 1.6. *POS annotation in the Penn Treebank [MAR 93]. For a color version of the figure, see www.iste.co.uk/fort/nlp.zip*

8 Obviously we did not participate in the original campaign, but we re-annotated part of the corpus for the experiments we led for [FOR 10].

9 We put them in blue here for easier reading.

10 In the *Penn Treebank*, these were tokens.

1.2.1.2. *Example 2: gene renaming*

Gene renaming annotation, on the other hand, implied annotating very few segments in the whole corpus (in average one renaming per file). The annotators had to identify the gene names involved in a renaming relation and annotate the former name of the gene and its new name (see Figure 1.7). Due to constraints imposed by the annotation tool, `Cadixe` [ALP 04], which was already in use when we joined the project, the annotators could not annotate the relation as such. The annotations in XML therefore included an identifier (<Former id="1">, <New id="1">), were added in-line and rendered in corresponding colors (one per renaming relation) by the tool. The corpus was not pre-annotated.[11]

> The yppB:cat and ypbC:cat null alleles rendered cells sensitive to DNA-damaging agents, impaired plasmid transformation (25- and 100-fold), and moderately affected chromosomal transformation when present in an otherwise Rec+ B. subtilis strain. The yppB gene complemented the defect of the recG40 strain. yppB and ypbC and their respective null alleles were termed recU and "recU1" (recU:cat) and recS and "recS1" (recS:cat), respectively. The recU and recS mutations were introduced into rec-deficient strains representative of the alpha (recF), beta (addA5 addB72), gamma (recH342), and epsilon (recG40) epistatic groups.

Figure 1.7. *Gene renaming annotation [JOU 11]. For a color version of the figure, see www.iste.co.uk/fort/nlp.zip*

1.2.1.3. *Example 3: structured named entities*

Finally, in the structured named entity annotation campaign, the work was done from scratch (no pre-annotation) on an advanced text editor (`XEmacs`), with a specific plug-in allowing the annotators to select the

11 We made some tests and pre-annotation did not really help as most gene names were not in a renaming relation, thus generating a lot of noise, and some could not be found by our pre-annotation tool (silence).

appropriate tags step by step following the structure of the tagset. For example, if the annotator selected the segment "Lionel" and the tag *pers*, the tool then proposed the subtypes *ind* or *coll* (for a theoretical illustration of the annotation, see Figure 1.8). Obviously, not all the text was annotated; however, it represented a much larger proportion than in the case of gene renaming. It should be noted here that the corpus was in French and some of it was transcribed speech (broadcast news) [ROS 12].

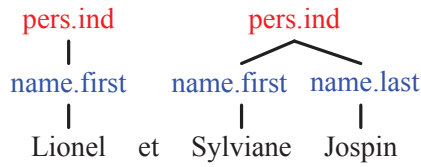


Figure 1.8. *Structured named entity annotation [GRO 11]*

These three annotation campaigns are so different that it seems difficult to compare them in terms of complexity. We will see that the complexity dimensions allow for that too.

1.2.2. *What to annotate?*

The first logical step in the manual annotation is to identify the segment of the signal to annotate. This "identification" consists, in fact, of two movements: (i) extracting, in a rather gross way, a piece to annotate from the signal (discrimination) and (ii) delimiting the precise boundaries of this segment.

1.2.2.1. *Discrimination*

If the second step is easy to grasp for most people, as delimitation builds on existing metrics like the word error rate (a well-known metric of the performance of speech recognition systems), the first step, the discrimination phase, is usually more difficult to understand and is often

overlooked. This dimension is important, as it captures the "needle in a haystack" effect, i.e. the fact that the segment to annotate is more or less easy to find in the source signal.

Let us consider examples 1 and 2. In POS annotation (example 1), all the tokens need to be annotated; there is nothing to search for (especially as the corpus was pre-annotated), so the discrimination will be null (0). On the contrary, in the gene renaming annotation case (example 2), the segments to annotate are scattered in the corpus and rare (one renaming per text on average), so the discrimination will be very high (close to 1).

When the segments to annotate are lost in the crowd of the text, i.e. when the proportion of what is to be annotated as compared to what could be annotated (resulting from the default segmentation, often token by token) is low, the complexity due to the discrimination effort is high. This is expressed in the following way:

DEFINITION 1.1.–

$$Discrimination_a(F) = 1 - \frac{|A_a(F)|}{|D_i(F)|}$$

where F is the flow of data to annotate, a is an annotation task, $|D_i(F)|$ is the number of units obtained during the segmentation of F at level i and $|A_a(F)|$ is the number of units to be annotated in the relevant annotation task.

Applying this metric, we obtain a discrimination of 0 for POS annotation and 0.95 for gene renaming.

1.2.2.2. *Delimitation*

Once the units are roughly identified, they have to be finely delimited. This is the delimitation process.

The definition of the delimitation metric is inspired by the slot error rate (an adaptation of the word error rate) [MAK 99]:

DEFINITION 1.2.–

$$Delimitation_a(F) = min(\frac{S + I + D}{|A_a(F)|}, 1)$$

where $|A_a(F)|$ is the final number of discriminated units, I is the number of inserted units, obtained by initial unit decomposition, D is the number of units deleted when grouping some of the initial units and S is the number of substitutions, i.e. the number of discriminated units that underwent a change in their boundaries other than that of the previous decomposition and grouping cases.

The delimitation complexity dimension is null in the case of gene renaming, as gene names are simple tokens. It reaches the maximum (1) for the structured named entity task, as many frontier changes have to be performed by the annotators from a basic segmentation in tokens.

The computation of both the discrimination and the delimitation complexity dimensions requires at least a sample of annotation, either from the campaign being prepared or from a previous, similar campaign.

1.2.3. *How to annotate?*

Once precisely identified, the units have to be characterized by the annotators. To do so, they rely on an annotation language with a certain expressiveness, instantiated in a tagset of a certain dimension.

1.2.3.1. *Expressiveness of the annotation language*

To evaluate the complexity due to the expressiveness of the annotation language, we decided to rely on an arbitrary (but logical) scale, graduated from 0.25 (type language) to 1 (higher order languages). Relational languages of arity 2 are attributed 0.5 in complexity and 0.75 is associated with relational languages with arity higher than 2.

In the simplest and most frequent case, the annotation language is a type language: annotating consists of associating a type with a segment of data. A lot of annotation tasks use this category of language: POS, speech turns, named entities, etc. The number of tags can vary, but this does not change the expressiveness of the language.

Establishing relations between units has become a relatively common task, but it is more complex. It requires us to connect different segments of data, which are often typed. The relations are often typed too and they can be oriented. This is for example the case for dependency syntax relations or gene remaining annotation.

In general, the relations are binary, but sometimes relations of arity above two are necessary, for example in information extraction: who bought what? when? to whom? at which price? In such cases, the annotation task is much more complex: the annotators have to discriminate, delimit and categorize the arguments of the relation, then to identify the couples, triplets, n-uplets of segments to annotate and finally, to label the relation.

Higher order languages are used when annotations are added to annotations, for example to qualify an annotation as uncertain. However, the complexity of this type of language is such that, in most cases, the problem is avoided by increasing the dimension of the tagset (creating a new feature associated with the main types).

Most annotation tasks correspond to a complexity of 0.25 or 0.5 for this dimension. In our examples, the POS and structured named entity annotation tasks are performed using simple type languages, so they reach a complexity of 0.25. Interestingly, the gene renaming campaign, that should correspond to 0.5 as it is a relation, reaches only 0.25 in complexity, due to the fact that the annotation tool did not allow for the annotation of real relations. Although it

simplified this complexity dimension, it made the tagset more complex to use.

1.2.3.2. *Tagset dimension*

The size of the tagset is probably the most obvious complexity dimension. It relates to short-term memory limitations and is quite obvious when you annotate. However, a very large number of tags is not necessarily a synonym for maximum complexity: if they are well-structured, like in the structured named entity annotation task (31 types and sub-types), then the annotators have to make choices from a reasonable number of tags each time, at different levels. In the structured named entity case (see Figure 1.9) they first have to choose between seven main types (*Person*, *Function*, *Location*, *Production*, *Organization*, *Time*, *Amount*), which corresponds to a degree of freedom of 6. Then, in the worst case (if they selected *Production*), they have to choose between nine sub-types, i.e. a degree of freedom of 8. Finally, sub-subtypes are available in some cases like *Location* and *Time*, so there can be a choice to make from a maximum of four tags, which corresponds to a degree of freedom of 3. We propose to use these degrees of freedom to compute the tagset dimension complexity, in order to take into account the fact that tagset constraints relieve the annotators from part of the categorizing effort.

The total degree of freedom ν for the choice of m labels is given by the following formula:

$$\nu \leq \nu_1 + \nu_2 + \ldots + \nu_m$$

where ν_i is the maximal degree of freedom the annotator has when choosing the i^{th} tag ($\nu_i = n_i - 1$).

The tagset dimension can then be computed using the following formula:

$$Dimension_a(F) = min(\frac{\nu}{\tau}, 1)$$

where ν is the global degree of freedom the annotator has when choosing a tag for an annotation task a within a flow of data F, and τ is the threshold from which we consider the tagset as arbitrarily large. In the experiments detailed below, τ is worth 50, based on the feedback of the annotators, but it can be adapted if necessary.

<table>
<tr><th colspan="3">Person</th><th colspan="3">Function</th></tr>
<tr><td colspan="2">pers.ind (individual person)</td><td>pers.coll (group of persons)</td><td colspan="2">func.ind (individual function)</td><td>func.coll (collectivity of functions)</td></tr>
<tr><th colspan="3">Location</th><th colspan="3">Production</th></tr>
<tr><td>administrative (loc.adm.town, loc.adm.reg, loc.adm.nat, loc.adm.sup)</td><td>physical (loc.phys.geo, loc.phys.hydro, loc.phys.astro)</td><td>facilities (loc.fac), oronyms (loc.oro), address (loc.add.phys, loc.add.elec)</td><td>prod.object (manufactured object)</td><td>prod.serv (transportation route)</td><td>prod.fin (financial products)</td></tr>
<tr><td colspan="3"></td><td>prod.doctr (doctrine)</td><td>prod.rule (law)</td><td>prod.soft (software)</td></tr>
<tr><td colspan="3"></td><td>prod.art</td><td>prod.media</td><td>prod.award</td></tr>
<tr><th colspan="3">Organization</th><th colspan="3">Time</th></tr>
<tr><td colspan="2">org.adm (administration)</td><td>org.ent (services)</td><td colspan="2" rowspan="2">time.date.abs (absolute date), time.date.rel (relative date)</td><td rowspan="2">time.hour.abs (absolute hour), time.hour.rel (relative hour)</td></tr>
<tr><th colspan="3">Amount</th></tr>
<tr><td colspan="3">amount (with unit or general object), including duration</td><td colspan="3"></td></tr>
</table>

Figure 1.9. *The tagset dimension: taking the structure into account in the structured named entity annotation task [GRO 11]*

Using these formulas, the tagset dimension complexity of the structured named entity annotation task reaches 0.34, which is quite low as compared to the 0.62 we would obtain without taking the structure into account.[12] As for the gene renaming annotation task, it involved only two tags, *Former* and *New*, but to bypass the annotation tool constraints, an identifier had to be added to disambiguate between the renaming couples that were present in the same text. As there were no more than ten renaming relations per text, this represents around 10 "subtypes", i.e. ν is close to 10 and the tagset dimension reaches 0.2, which is not so far from the result for structured named entities.

12 These results differ from the ones presented in [FOR 12d] because we simplified the example (the annotation task also included components).

Both the expressiveness of the language and the tagset dimension can be computed *a priori*, without any annotation done yet, provided the tagset has been defined.

1.2.3.3. *Degree of ambiguity*

Disambiguating the units to annotate is at the heart of the work of the annotators. This is obviously a complexity dimension and this is where most of the interpretation lies, but it is very difficult to evaluate precisely. However, we propose two ways of approximating it.

1.2.3.3.1. Residual ambiguity

First, we can observe the traces left by the annotators when they are given the opportunity and the possibility to do so. For example, in a gene and protein names annotation campaign [FOR 09], we gave the annotators the possibility to add an uncertainty feature to the annotation (see Figure 1.10). Although one of them used this possibility, it is quite useful to evaluate the ambiguities they faced.

> [...] <EukVirus>3CDproM< /EukVirus> can process both structural and nonstructural precursors of the <EukVirus **uncertainty-type** = "too-generic"><taxon>poliovirus< /taxon> polyprotein< /EukVirus> [...].

Figure 1.10. *Example of typed trace left by the annotator when annotating gene and protein names [FOR 09]*

We call this the *residual ambiguity* and we define it in a very simple way:

DEFINITION 1.3.–

$$Ambiguity_{Res,a}(F) = \frac{|Annot_A|}{|Annot|}$$

where a and F are the annotation task and the flow of data to be considered and where $|Annot_A|$ and $|Annot|$ are respectively the number of annotations bearing an ambiguity mark and the total number of annotations added to F.

By definition, the residual ambiguity can only be computed from an annotation sample, if the possibility to add traces was given to the annotators. In the gene renaming campaign, it was nearly null (0.02), probably due to the fact that, again, only one annotator added traces. This metric is not completely reliable and it should be associated with another one whenever possible.

1.2.3.3.2. Theoretical ambiguity

The second way to measure the complexity of the disambiguation process is to measure the degree of *theoretical ambiguity* for the tasks where several occurrences of the same vocable are annotated. This applies to POS annotation or semantic disambiguation, but not to gene renaming annotation.

This metric relies on the idea that ambiguous vocables are annotated with different tags in different places in the text (or flow of data). We then need to compute the proportion of the units to annotate which correspond to ambiguous vocables, taking into account their frequency. This can be done using the following formula:

DEFINITION 1.4.–

$$Ambiguity_{Th,a}(F) = \frac{\sum_{i=1}^{|Voc(F)|} (Ambig_a(i) * freq(i,F))}{|Units_a(F)|}$$

with

$$Ambig_a(i) = \begin{cases} 1 \text{ if } & |Labels_a(i)| > 1 \\ 0 \text{ else} \end{cases}$$

where Voc *is the vocabulary of the units of the flow of data* F*,* $|Voc(F)|$ *the size of the vocabulary,* $freq(i, F)$ *the frequency of the vocable* i *in* F*,* $|Units_a(F)|$ *the number of units to annotate in* F *and* $|Labels_a(i)|$ *the number of tags available for the vocable* i *for the annotation task* a*.*

Again, to compute this metric, we need an annotation sample or results from a similar task.

1.2.4. *The weight of the context*

The context to take into account during annotation is an obvious complexity factor. However, this dimension is not independent of all the above-mentioned dimensions. It directly influences the discrimination, delimitation and disambiguation processes, as the larger the context, the more difficult it gets to identify the units to annotate and to disambiguate them. Nonetheless, we decided not to include it as a modifying factor of these three dimensions, first to keep them simpler, and second because of its strong identity.

In NLP, the context is traditionally the co-text taken into account by the systems. Despite some evolution (particularly in discourse annotation or semantic annotation like football), the sentence is still the favored processing unit in our domain. However, for the annotators, the context is not only the text they have to read to be able to annotate (identify and characterize) properly, but also the knowledge sources they need to consult. These sources usually include the annotation guidelines, but they may also be external sources, either identified during the campaign preparation, like nomenclatures *à la* `SwissProt`,[13] or be found by the annotators themselves, on the Web or elsewhere.

13 See: http://www.uniprot.org/.

Obviously, the more accessible and predictable the source, the less complex it is for the annotators to get the piece of information they need. As for the co-text, the larger the context to take into account, the more complex it is to annotate (see Figure 1.11).

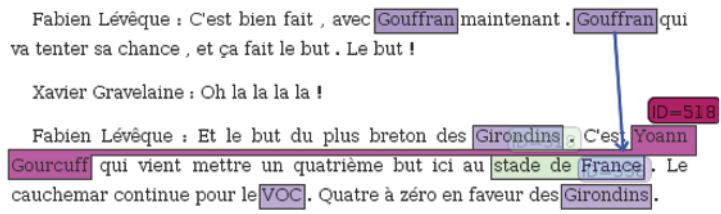


Figure 1.11. *Example of annotation of a goal in football annotation [FOR 12b]: a context of more than the sentence is needed*

We therefore designed a common discreet scale including both these sub-dimensions. In this scale, 0 corresponds to an impossible case, where there is no need for an annotation guide and no co-text to take into account. This should never happen, as the consensus has to be somehow transmitted to the annotators. 0.25 corresponds to a case where an annotation guide is needed OR the immediate co-text is needed to annotate. Logically, the complexity reaches 0.5 when the OR of the previous description changes to an AND, i.e. when the annotators need guidelines AND a small context to annotate. Another case in which we reach 0.5 is when a larger part of the data (like the sentence) OR an identified external source of knowledge is needed. 0.75 corresponds to the case when the annotators need to read a larger co-text AND have to consult an identified external source. It also covers the cases in which the annotators have to access unpredicted sources of knowledge OR have to read the whole text to be able to annotate. Finally, 1 is for cases where the annotators both have to consult previously unidentified sources of knowledge AND the whole data flow (usually, text).

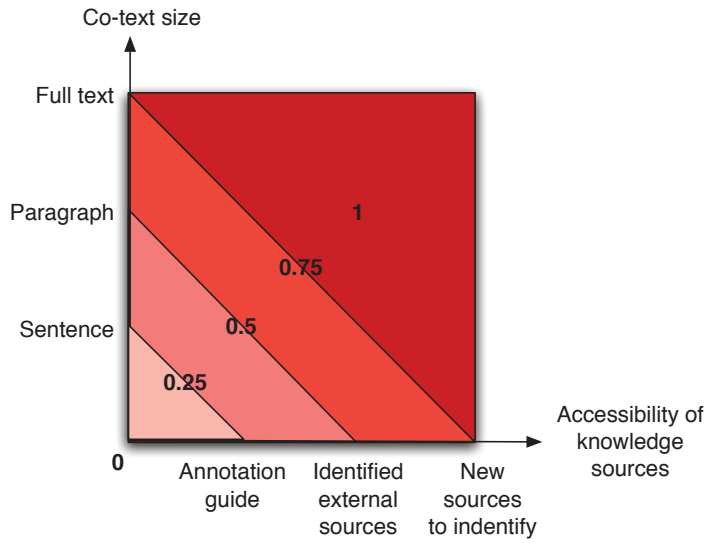


Figure 1.12. *The context as a complexity dimension: two sub-dimensions to take into account*

The gene renaming task is very complex from that point of view (1), as it required the annotators to read the whole text and they sometimes needed to consult new external sources. POS annotation would be close to 0.5, as most of the time only the guidelines and a small co-text are needed to annotate.

1.2.5. *Visualization*

Once the 6 complexity dimensions are computed, it is rather easy to put them into a spiderweb diagram to visualize the complexity profile of the annotation task. This type of representation can prove useful to compare the complexity of different tasks. Figures 1.13 and 1.14 present examples of what can be obtained applying the complexity grid, even in a fuzzy way (for the POS annotation task).

In the *Penn Treebank* POS annotation task, the corpus was pre-segmented and pre-annotated, so the discrimination and

delimitation are null. The annotation language is a type language. The tagset contains 36 tags [SAN 90], so ν equals 35, but if we consider that there is an implicit structure in the tagset, with *JJR* and *JJS* being subtypes of *JJ*, then $\nu = 20 + 5 = 25$ and the complexity dimension of the tagset is 0.5. The annotation guidelines allowed for the usage of an ambiguity mark (a vertical slash, "|") in case of true ambiguities, so even if this is not exactly residual ambiguity, it can still be computed. However, for the *Wall Street Journal* part of the corpus, it represents only one case, so it probably can be considered as null over the whole corpus. As for the theoretical ambiguity, Dan Jurafsky and James H. Martin, in the new edition of their well-known book [JUR 09] evaluate the ambiguity in POS for English saying that "[...] the ambiguous words, although accounting for only 14–15% of the vocabulary, are some of the most common words of English, and hence 55–67% of word tokens in running text are ambiguous".[14] This implies that the theoretical ambiguity is rather high and without even computing it precisely, we can evaluate it at 0.5. The context to take into account is restricted to an annotation guide and a limited co-text (0.5).

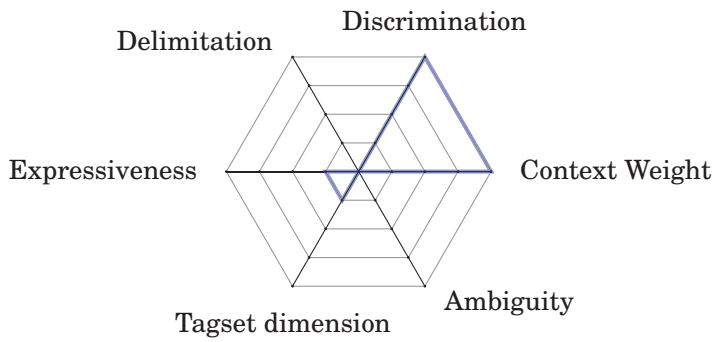


Figure 1.13. *Instantiated visualization: the delimited surface represents the complexity profile of the annotation task, here, gene renaming*

14 See the draft here: https://web.stanford.edu/jurafsky/slp3/9.pdf.

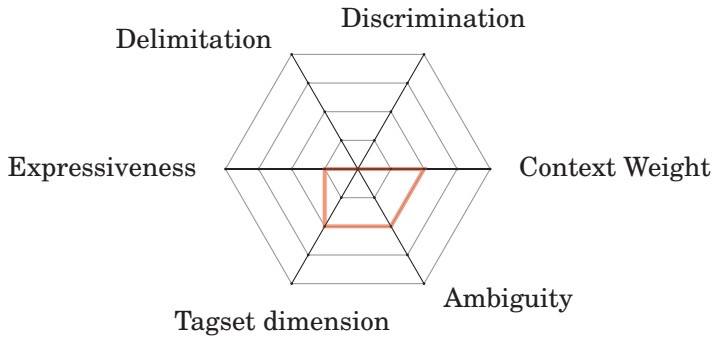


Figure 1.14. *Instantiated visualization: POS annotation in the Penn Treebank*

The complexity profiles of these annotation tasks are very different, thus reflecting the need for very different solutions to limit the complexity of the tasks. For POS annotation, even without pre-annotation, the discrimination and delimitation would have been low, due to the fact that, in this campaign, only tokens were annotated. However, the tagset dimension complexity could have been reduced by structuring the tagset more (and taking this structure into account in the annotation tool). As for the gene renaming campaign, it could have benefited from an "intelligent" pre-annotation (taking into account keywords like "renamed") to reduce the discrimination effort. It could also have been easier from the context point of view if a precise list of the sources to consult were provided in the guidelines.

1.2.6. *Elementary annotation tasks*

We saw that the gene renaming annotation task can be analyzed with the complexity grid as it was performed in the campaign, with identifiers as features in the XML tags. However, it should probably have been annotated differently, with a more suitable tool and with real relations. In this case, it would have been difficult to analyze it as a whole.

We propose to decompose such tasks into *Elementary Annotation Tasks* (EATs) and to compute the complexity of the various EATs independently, the global complexity of the task being a combination of the local EATs' complexity. Note that EATs do not necessarily correspond to annotation levels or layers [GOE 10] or to the practical organization of the work.

DEFINITION 1.5.– *An Elementary Annotation Task (EAT) is a task that cannot be decomposed. We consider that an annotation task can be decomposed into at least two EATs if its tagset can be decomposed into independent reduced tagsets. Tagsets are independent when their tags are globally compatible (even if some combinations are not allowed), whereas the tags from a unique tagset are mutually exclusive (apart from the need to encode ambiguity).*

In the gene renaming campaign, for example, the annotation of the relations can be analyzed as a combination of two EATs: (i) identifying gene names in the source signal and (ii) indicating which of these gene names participate in a renaming relation. The two tagsets are independent and the global task is easier to analyze in the following way.

1.2.6.1. *Identifying gene names*

Only a few words are gene names, so the discrimination is high (0.9). Gene names, in our case, are only tokens, so delimitation is null. The tagset dimension is null too, as there is only one tag (*gene name*). We use a type language (expressiveness=0.25). The ambiguity is very low, as only few gene names are ambiguous and the annotators left little trace of uncertainty on this. On the contrary, the necessary context is relatively high (between 0.5 and 0.75), as although only a few words are needed to identify a gene name, the annotators sometimes had to consult external sources.

1.2.6.2. *Annotating gene renaming relations*

This EAT consists of identifying, among all the gene name couples appearing in the same text (*PubMed* abstracts), the ones that are connected by a renaming relation, i.e. the ones that are the former and the new names of one gene. As we already said, renaming relations are rare, so the discrimination for this EAT is high (0.95). Gene names are already annotated (EAT 1), so delimitation is null. The relation is oriented, but there is only one type of relation, so the tagset is close to null. The annotation language is relational (0.5) and ambiguity is very low according to the traces left by the annotators (0.02). The context is maximum (1), as the annotators had to read the whole text to be able to identify renaming relations and they at least had to consult identified external sources.

The two EATs are then combined to provide a global view on the campaign with a scale that is twice the scale for one EAT (see Figure 1.15). In this particular case the result is very close to that of the single EAT analysis (compare with Figure 1.13).

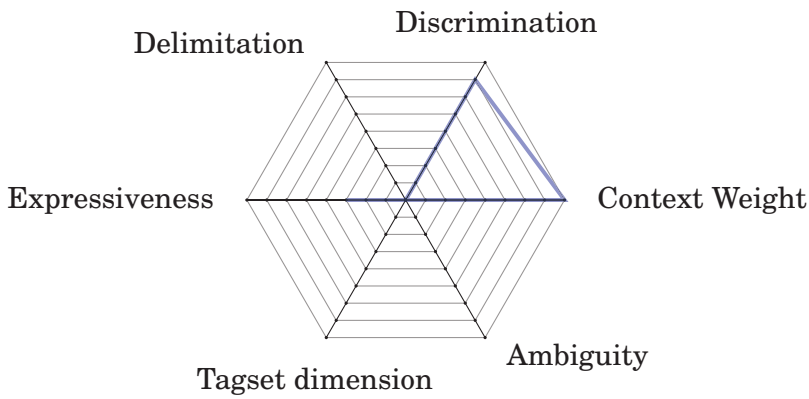


Figure 1.15. *Synthesis of the complexity of the gene names renaming campaign (new scale x2)*

Note that the decomposition into EATs does not imply a simplification of the original task, as is often the case for Human Intelligence Tasks (HITs) performed by *Turkers* (workers) on `Amazon Mechanical Turk` (see, for example, [COO 10a]).

1.3. Annotation tools

Once the complexity profile is established, the manager has a precise vision of the campaign and can select an appropriate annotation tool.

Annotation tools make manual annotation much easier, in particular when using markup languages like XML. These interfaces allow us to avoid the tedious writing of tags and the associated typing errors, but their contribution reaches far beyond that.

If there are many articles detailing a specific annotation tool, only a few provide a high level view on the subject. To our knowledge, only [DIP 04, REI 05] and [BUR 12] present an in-depth comparison of the tools in order to allow for their evaluation. However, these articles only consider a limited number of annotation tools (five in [DIP 04], two in [REI 05] and three in [BUR 12]) and the analysis carried out in the first two is focused on a specific annotation task (purely linguistic annotation in the first one and video annotation for the second one). The present state of the art uses some information from these articles, but it is mainly drawn from our own experience and analysis of the existing tools (a non-exhaustive list of these tools is presented in Appendix).

1.3.1. *To be or not to be an annotation tool*

Before going into more detail about annotation tools, we need to clarify what they are:

DEFINITION 1.6.– *A system supporting manual annotation, or annotation tool, is an interface facilitating the manual annotation of a signal.*

Some tools support the manual annotation of non-textual corpora, like video (Anvil[15] or Advene[16]), speech (Praat[17]) or music (wavesurfer[18]), but such an inventory would take us too far. We therefore restrain our analysis to manual text annotation.

We do not present Web annotation interfaces either, but their features are usually close to that of the tools we present here, without being as complex and rich. Finally, we do not consider XML or text editors as annotation tools as such. As this can seem surprising for some, we will explain why.

A number of manual annotation campaigns use XML editors to help the annotators in their work. This was the case for example for the manual annotation of Stendhal's manuscripts [LEB 08], which was performed using Morphon.[19] Another example is the *Definiens* project of annotation of the definitions of the French dictionary *Trésor de la Langue Française* [BAR 10], in which the annotators used oXygen.[20] Because we had to use XML tags, a partner imposed on us to use Epic[21] to annotate patents in pharmacology.

An XML editor is designed to edit XML files, not to annotate. If the features seem similar, the underlying logic is

15 See: http://www.anvil-software.de/.

16 See: http://liris.cnrs.fr/advene/.

17 See: http://www.fon.hum.uva.nl/praat/.

18 See: http://sourceforge.net/projects/wavesurfer/.

19 This tool is now deprecated, see: https://collab.itc.virginia.edu/wiki/toolbox/Morphon's.

20 See: https://www.oxygenxml.com/.

21 Now PTC Arbortext Editor: http://www.ptc.com/service-lifecycle-management/arbortext/editor.

quite different: an XML editor supports the modification of an XML file, not the annotation of a corpus of texts. The first difference concerns the notion of corpus, which does not exist in XML editors. This prevents us from having a global vision of the annotation. Moreover, these tools usually do not support standoff annotation, which prevents the easy annotation of overlaps, discontinuous groups and relations. Obviously, the management of annotation campaigns is not supported by such tools. Finally, some annotation tools (like `Knowtator`, `Glozz`, `Slate` or `GATE`) allow us to visualize the disagreements between annotators and for some of them to compute the inter-annotator agreement. This is never possible with an XML editor.

Text editors present the same limitations. In addition, they usually do not provide any means to validate the XML and annotators may therefore put the tags in the wrong order. However, simple tools can prove very useful for limited experiments (for example in prototyping) or when they are completed by normalization scripts.

Given the multiplication of annotation campaigns, it would take months to install and test all the annotation tools which are in use today. They are more or less available, more or less maintained, some are open-source, some not. From our point of view, the lifespan of a tool depends on the same criteria as that of corpora as described in [COH 05]: an annotation tool, to be used on the long-term, should be freely available, maintained and well-documented. This means that to survive on the long run, annotation tools like any software should be supported, either by a community of developers or by an institution. This kind of tool should also be easy to install and use. Ergonomics is important, as features which are difficult to access are not used by the annotators. We witnessed this in an annotation campaign in microbiology, in which the annotators often failed to report their uncertainties (which are needed to compute the residual ambiguity, see section 1.2.3.3) because the corresponding feature was not easy to add.

1.3.2. *Much more than prototypes*

Annotation tools are generally designed for one or several annotation tasks, rather than around the needs of the annotators. However, the tendency seems to evolve towards taking them more into account, through more user-friendly and more efficient interfaces. In addition, there is a growing consensus about the use of XML and standoff annotation, which seems to correspond to a form of standardization of the formalisms.

1.3.2.1. *Taking the annotators into account*

Even if there is still room for improvement (there are no keyboard shortcuts in `Glozz` [WID 09] and too many windows in `MMAX2` [MÜL 06], etc.), the interfaces of the annotation tools are becoming more and more user-friendly. For example, they offer editing possibilities that allow the annotators to automate some tasks (*annotate all* in `GATE` [CUN 02], `Glozz` and `Djangology` [APO 10], automatically generated regular expressions in `SYNC3` [PET 12], rapid selection with one selection in `Knowtator` [OGR 06]). They also often allow us to hide some annotations (by levels, or according to other criteria, like in `Glozz`) to ease the visualization and almost all of them allow for a certain level of customization (at least the colors of the tags).

Moreover, searching and editing annotations is sometimes made easier, thanks to powerful search engines, which allow us to search both the text and the annotations (in GlozzQL for `Glozz`, using regular expressions in `UAM CorpusTool` [O'D 08]). Once the annotators are trained for the task and with the annotation tool, they annotate more rapidly and more comfortably [DAN 09].

It has to be noticed that, even if it is not an equally shared preoccupation for all the annotation tools developers, the vast majority of the available interfaces are written in Java and therefore support most natural languages.

Some features, even if they are not yet widespread, are quite useful. For example, brat [STE 11] associates with each annotation a unique URL, which allows us not only to link the data over the Web, but also to unambiguously reference an annotation in the documentation. Glozz, which was designed for discourse annotation, offers a global visualization of the text that helps annotating macro level structures.

However, some useful features are still missing. For example, in a campaign where old press was annotated with named entities [ROS 12], we noted that the digitalized text was often erroneous and that the annotators needed to see the original image to be able to annotate correctly. For such cases, it would be interesting to include the original scanned source as an image into the tool. Another issue is that of meta annotations. If some tools like Glozz allow us to add commentaries or uncertainties, these have to be planned beforehand in the data model and are not proposed by default, even though they are essential.

Finally, most annotation tools are not robust enough and are not suitable for the annotation of large files (this is in particular the case for Glozz and GATE).

1.3.2.2. *Standardizing the formalisms*

XML has become the most widely used export and storage format for annotations, associated with standoff annotation in most tools. Annotations are standoff when they are presented separately from the source signal, often in another file. This evolution was advocated in [LEE 97, IDE 06]. Standoff annotation presents many advantages: it preserves the source corpus (the rights on this corpus are respected and it cannot be polluted) and it allows us to annotate discontinuous groups, overlaps, inclusions and relations

(oriented or not). It is also more flexible than inline annotation since new annotation levels can be added, without modifying the existing ones. Finally, each annotation level can be manipulated separately, including in different files, in particular to compare annotations.

However, some annotation campaign managers prefer to give the annotators the possibility to access the source data. This was in particular the case in the structured named entity campaign in which we participated in [ROS 12]. In this case, the disadvantage is that an annotator may introduce errors in the file, for example by inserting XML tags in the wrong order, thus transforming it into an invalid XML file. `GATE` is the only tool we know of that offers the possibility to modify the source data as an option, leaving to the manager the choice of whether or not to allow such modifications.

The possibility to annotate relations (oriented or not) or sets (in the case of anaphora) is becoming more and more commonly proposed. However, `Callisto` [DAY 04] and `GATE` offer limited capabilities and `UAM CorpusTool`, `Cadixe` [ALP 04] and `Eulia` [ART 04] do not support this.

Some tools allow for the definition and usage of different annotation layers (`MMAX2`, `Glozz`, `UAM CorpusTool`), corresponding to linguistic levels (POS, syntax, etc.) or to "groups", which are defined by the designer of the data model, like in `Glozz`. The flexibility of the definition of these groups allows for the grouping of semantically close elements, without them having any specific linguistic meaning (like *Player*, *Team*, *Referee* and *Coach* in football matches annotation [FOR 12b]). These groups can then be used to annotate (one group being annotated before the other), to customize the display (hiding or not a group) and for the inter-annotator agreement computation.

1.3.3. *Addressing the new annotation challenges*

We observe that annotation tools are progressively reaching maturity and are evolving in three main directions: genericity, collaboration and campaign management.

1.3.3.1. *Towards more flexible and more generic tools*

We have witnessed, in the last decade, an evolution from task-oriented annotation tools towards more generic and more flexible tools, often using plug-ins (this is the case for example in GATE) or a common Application Programming Interface (API) (like in the LDC tools [MAE 04]).

This genericity, when it results from an evolution of a task-oriented tool or of a tool with a different objective, often generates complexity issues and the final tool is often difficult to install and to parameterize. This is the case, in particular, for GATE and Callisto, whose underlying logics are not so easy to understand (GATE, for example, was not originally designed for manual annotation). This results in a long learning curve for the campaign manager. When they are designed to be generic manual annotation tools from the very start, they are usually easier to get into and to parameterize. This is the case, for example, for WebAnno (see Appendix, section A.4.4) and CCASH [FEL 10].

Moreover, the generalization of XML allows us to adapt more easily to *de facto* standards like TEI (Text Encoding Initiative, a format that is well-known in the humanities), and thus to share annotated data.

Finally, some annotation tools designed for biocuration, like brat, include linked data annotation capabilities, which allow us to normalize the annotations using a unique identifier and to share them. This feature is very powerful and the development of linked data will accelerate its generalization. For the moment, linguistic linked data are still limited, but the development of new formats like NIF (NLP Interchange Format) [HEL 13] will certainly help.

1.3.3.2. *Towards more collaborative annotation*

As we saw in section 1.1, annotating implies to reach a consensus on the definition and perimeter of the used categories. This consensus is at the heart of the annotation process and it cannot be built without collaborating. Annotation is therefore by essence collaborative. However, we hope that we show in this book that collaboration in annotation can take various forms.

Georgios Petasis uses the adjective *collaborative*/*distributed* in [PET 12] to distinguish between collaborative annotation tools as Web applications and his tool, which is not a thin client. By doing so, he is trying to unravel two terms that are often mixed up today. The term *collaborative annotation* is ambiguous and for some means crowdsourcing annotation, for others annotation by a community of experts, if not both (the call for paper for the Linguistic Annotation Workshop VI is a good example of that),[22] and for others, including Petasis, it means the participation in a common annotation project.

Collaboration in annotation is defined along two axes: its visibility to the annotators (do they know they are collaborating?) and the potential means used for its implementation, as collaboration can be direct or indirect. For example, the wiki-like annotation mode, in which each annotator sees what the others are doing and can modify their annotations, like in `brat` and as an option in `SYNC3`, is fully collaborative, as it is both direct and visible to the annotators.

On the contrary, the adjudication by an expert (of the field of the annotation) of annotations added by others is a form of

22 *"The special theme for LAW VI is Collaborative Annotation (both community-based and crowd-sourced)"*: http://faculty.washington.edu/fxia/LAWVI/cfp.html

indirect collaboration, since the expert benefits from the work previously done and is inspired by it. Beyond that, annotating in parallel parts of the corpus and using the resulting inter-annotator agreement to improve the annotation guidelines, which in turn will be used by the annotators, is another form of collaboration since the work carried out by some upstream influences the annotation to be performed by all. This type of indirect collaboration is rather invisible to the annotators, as they only see its negative manifestation (when they are told about it): their disagreements.

A more obvious form of collaboration is the possibility to interact with other annotators and to keep track of these interactions. If in `EasyRef` this interaction is indirect through bug reports [CLE 08], it is direct and clearly visible in `AnT&CoW` [LOR 06][23] as a forum. Another benefit of this type of interaction is that it fosters the motivation of the annotators. Unfortunately, it is not yet offered in existing annotation tools for NLP. We plan to add such a feature to the Game With A Purpose (GWAP) `ZombiLingo` [FOR 14c], so that the annotators can both socialize and correct themselves.

Collaboration has always been part of annotation. However, we have been witnessing, since the advent of the Web 2.0, the development of new forms of collaboration. We present in details, in the second part of this book, the various forms of crowdsourcing annotation, but we can summarize here the main forms of collaboration it implies. Games with a purpose like `Phrase Detectives` [CHA 08] or `ZombiLingo` usually provide for an indirect and visible collaboration (through the obtained scores and the leaderboards).[24] As for the microworking platforms like `Amazon Mechanical Turk`, they only allow for a very indirect collaboration

23 `AnT&CoW` is not an annotation tool for NLP, which is why it does not appear in the Appendix.

24 Some future features of `ZombiLingo` will allow for more direct collaboration.

through the agreement (or disagreement) among the workers, which is invisible to them (they have very little feedback on their work).

This evolution is accompanied by a raising awareness of the importance of the annotators' training and of the evaluation of the annotation. Both Phrase Detectives and ZombiLingo put emphasis on these two points, with mandatory training and regular evaluations of the performance of the annotators. One of the objectives of collaboration is to facilitate the training of the annotators thanks to co-evaluation.

1.3.3.3. *Towards the annotation campaign management*

To our knowledge, the first research paper to explicitly mention annotation campaign management is [KAP 10], which presents SLATE, a tool that offers features not only to support the annotation process, but also, and this is what makes it original, a more macro vision of the annotation process, including a clear definition of its actors (*administrator* and *annotators*, considered as completely distinct). Thanks to SLATE, the administrator can distribute and monitor the texts to annotate and therefore manage the corpus. The corpus itself is versioned throughout the project and each annotation is identified with the version number of the project, which also corresponds to that of the tagset at the time of the annotation. SLATE also includes comparing and merging features.

A more formal and explicit definition of the roles can be found in GATE Teamware [BON 10], which identifies three of them (*campaign manager*, *editor* or *curator* and *annotator*) and in WebAnno [CAS 14] (*users*, *curator* and *administrators*). Egas distinguishes only between *managers* and *curators*. As for the annotation management features, they are similar in Djangology and GATE Teamware and were to be developed in CCASH and Callisto (but it does not seem to be done yet).

The evolution towards annotation management is now obvious. However, it started long before 2010. Interfaces that allow us to compare annotations and to compute inter-annotator agreements were added in many tools (Knowtator, MMAX2, Glozz, SYNC3). Besides, if NLP platforms like GATE propose automatic processing to optimize manual annotation, most of the other tools support the condition that such processing be applied beforehand (provided the result is adapted to the format of the tool, like in Glozz) and some even provide some pre-annotation, like *tag dictionary* (a unit is pre-annotated with the tags that are associated with it earlier in the corpus), in Djangology and CCASH.

Given the potential biases generated by pre-annotation [FOR 10], we consider that automatic processing should be decided upon and applied by the campaign manager. It therefore falls under campaign management and not annotation as such. The same goes for the possibility to modify the annotation schema during the campaign (this is proposed in UAM CorpusTool, GATE and ANALEC [LAN 12]).

Finally, tools like Slate or EasyRef[25] propose to define constraints on the annotation (for example, in EasyRef, pop up menus allowing only for the actions authorized in this context), which, again, should be defined by the manager.

The monitoring of the annotation campaign is another feature offered by many "simple" annotation tools that is directly linked to campaign management, even if it can be useful to annotators too. For example, brat can be configured to monitor the time spent by an annotator on a document and on each editing and typing action (a similar feature is proposed in CCASH). EasyRef keeps track of the activities on the system using logs. This monitoring, which is done locally

25 This tool is mentioned here because it offered interesting original features, but it was used only in one annotation project.

in annotation tools, is enriched by a more global management in annotation management tools like WebAnno, SLATE or GATE Teamware, which allows us to visualize the progress of the campaign and of the annotators. However, this feature requires that the notion of a corpus is taken into account, which is not the case in all annotation tools (it is for example absent in the annotation part of Glozz).

This evolution towards annotation management goes hand in hand with the multiplication of Web-based tools (WebAnno, Slate, Egas, etc.). This presents many advantages, in particular it offers the possibility to work from distance, but it can also be troublesome, for example for under-resourced languages annotation, as the annotators may have a limited Internet access.

1.3.4. *The impossible dream tool*

The rising diversity in annotations (see section I.2.1) implies a variety of annotation tools. From text to video or speech, from the micro level (POS annotation) to the macro level (discourse), a unique, universal annotation tool, which would satisfy the needs and constraints (for example, the preservation of the original format) of each and everyone seems inconceivable.

In addition, many annotation campaign managers would rather develop a new tool, adapted to the constraints of their campaign and which can be as simple as an old school Emacs plugin, than try and adapt to an existing tool, which would be time-consuming, could bias the annotation due to intrinsic limitations, and might in the end be disappointing.

If some tools are more used than others, often because they are well-featured and maintained (this is for example the case for GATE and WebAnno, and to a lesser extend, for Glozz and brat), there is yet, as of today, no annotation tool

winning unanimous support. Developing a generic, reliable and well-documented annotation tool is a long-term endeavor. For example, it took two persons six months only to conceive Glozz and the same time to develop it.[26]

In addition, if there are many annotation tools available today, only a few of them provide features that allow to manage an annotation campaign. To our knowledge there are only a couple of them: Slate, GATE Teamware, Djangology, WebAnno and Egas [CAM 14]. Moreover, two of them present important limitations: Djangology is not maintained anymore and Egas is solely provided as an online service, specializing in biocuration. Finally, none of them propose any feature to prepare the campaign (see section 1.1). They provide no means to anticipate the complexities of the annotation campaign (see section 1.2) and to select the appropriate automation or inter-annotator metric to apply (see section 1.4). The analysis of complexity also provides useful information to select the most appropriate inter-annotator agreement metric.

The analyses of the annotation process and complexity dimensions presented in this chapter are therefore useful complements to your favorite annotation tool when preparing an annotation campaign.

1.4. Evaluating the annotation quality

1.4.1. *What is annotation quality?*

To be considered as "good", an annotation has to be valid, i.e. the notes added to the source have to be of the correct type and associated with the right segment in the flow of data. However, manually annotating is by definition interpreting, therefore there is no such thing as a "(ground) truth". We cannot directly measure the validity of manual annotation, we can only measure its reliability, i.e. how

26 Yann Mathet, personal communication, January 12th, 2011.

consistent the annotators were in annotating. This reveals how well they assimilated the guidelines and how coherent these guidelines are.

This *reliability* can only be evaluated by computing the agreement between annotators, or inter-annotator agreement, which is obtained by comparing the annotations of the same text made by different annotators. In addition to the inter-annotator agreement, which allows us to measure the *stability* of the annotation, the agreement of the annotator with him or herself later in the campaign (the intra-annotator agreement) also needs to be computed, in order to capture the *reproducibility* of the annotation [GUT 04].

If computing the intra- and inter-annotator agreements is essential, it does not have to be done on the whole corpus, for obvious reasons of cost-effectiveness. However, we strongly advise to do this very early in the campaign, so as to identify and address the problems rapidly, as was done in [BON 05].

Finally, to complete the quality evaluation, it is essential to randomly check the annotations on which the annotators agree. In Składnica , a Polish treebank, 20% of the agreed annotations were in fact wrong [WOL 11].

The inter-annotator agreement research field has been very active in the past decade and is still evolving rapidly. We present here the main issues and metrics and refer the reader who would like to go further to more detailed articles, in particular [ART 08] and [MAT 15].

1.4.2. *Understanding the basics*

1.4.2.1. *How lucky can you get?*

The most obvious and simplest metric measuring the inter-annotator agreement is the *observed agreement* (A_o). It

corresponds to the percentage of times the annotators agreed, i.e. the number of agreeing annotations times 100 over the whole number of annotations. This metric is very simple and easy to compute, but it should not be used as such as it does not take chance into account. Why is that important?

In order to demonstrate the influence of chance on the inter-annotator agreement results, let us take a very simple example.[27] In an annotation task involving two categories and no segmentation (like the two sides of a coin), two annotators who would pick any of the two categories randomly (like tossing the coin) would statistically agree half of the time ($A_o = 0.5$). Therefore, in this case, an observed agreement below this baseline would be very bad (worse than by chance). The main issue with this kind of metrics is that their real scale depends on the context of the campaign: the minimum that can be obtained by chance differs according to the number of categories and annotators. This makes the results very difficult to interpret.

But it can be worse. In the same case (two categories, A and B, and predefined segments) but with three annotators, it is impossible for them to completely disagree ($A_o \neq 0$): if Annotator 1 says A and Annotator 2 says B, Annotator 3 will necessarily agree with one of the first two annotators. So the observed agreement will at least be 0.33, even before taking chance into account (see Table 1.16).

Pairs	Annotations	Agreement
Annotators 1&2	A B	No
Annotators 1&3	A A	Yes
Annotators 2&3	B A	No

Figure 1.16. *Case of impossible disagreement, with 3 annotators and 2 categories*

27 This example was suggested to us by Yann Mathet, from GREYC-CNRS (personal communication, Fall 2015).

Now, let us make a detour and consider the "truth". If the right answer is A, then they succeed in 4 out of 6 times, so they are right 66% of the time. But if the right answer is B, then they succeed in 2 out of 6 times, so they are right 33% of the time. Finally, there can be a perfect inter-annotator agreement ($A_o = 1$), for example if the three annotators say A, and 0% truth (if the right answer was B). On the contrary, 100% success in finding the truth implies a perfect agreement.

The same campaign with only two annotators allows for a total disagreement. In one case (3 annotators) the scale begins at 0.33 and in the other (2 annotators), it starts at 0, without even taking chance into account.

1.4.2.2. *The kappa family*

As of today, the reference article on the subject of this family of inter-annotator agreement metrics is the one written by Ron Artstein and Massimo Poesio in 2008 [ART 08]. Its presents in details and very clearly these coefficients. We will focus here on the two most well-known, Scott's pi [SCO 55] and Cohen's kappa [COH 60]. These coefficients are applicable to two annotators only, but generalizations to more than two annotators are available, like Fleiss' kappa [FLE 71], a generalization of Scott's pi, or multi-κ, a generalization of Cohen's kappa [DAV 82].

Pi and kappa are computed from the observed agreement (A_o), but they take chance into account, which is represented in the *expected agreement* (A_e). Hence, the metrics are defined using the same formula:

$$\kappa,\pi = \frac{A_o - A_e}{1 - A_e}$$

The only element that differs is the way they evaluate chance, i.e. the expected agreement (A_e). In one case (pi), the categories are affected to units by chance mimicking the way they were actually affected by the annotators, but the

annotators themselves are supposed to behave in the same way (their behaviors are averaged). In the other case (kappa), both the categories and the annotators can by chance behave according to the way they behaved in reality.

1.4.2.2.1. Scott's pi

This coefficient is also called K in [SIE 88] or Kappa in [CAR 96] (or Carletta's kappa). In pi, the distributions realized by chance by the annotators are equivalent, but the chance distribution of the units (u) between categories (k) is not homogeneous and it can be estimated by the *average* distribution generated during their annotation by the annotators. The expected agreement for pi (A_e^π) is therefore defined as follows, with n_k being the number of units annotated with k by the two annotators.

$$A_e^\pi = \sum_{k \in K} (\frac{n_k}{2u})^2$$

1.4.2.2.2. Cohen's kappa

This coefficient models chance by hypothesizing that the distribution of units between categories can differ from one annotator to an other. In this case, the probability for a unit (u) to be affected by a category (k) is the product of the probability that each annotator assigns it in this category. The expected agreement (A_e^κ) is therefore defined as follows n_{c1k} being the number of assignments to k for annotator 1:

$$A_e^\kappa = \sum_{k \in K} \frac{n_{c1k}}{u} . \frac{n_{c2k}}{u}$$

Note that, by definition, $\pi \leqslant \kappa$. Usually, κ and π give very close results [DIE 04], which means that there is little bias between the annotators. It is therefore useful to compute both coefficients to check that.

1.4.2.3. *The dark side of kappas*

The coefficients of the kappa family are very efficient, they take chance into account and are not so difficult to compute. For these reasons, they have been widely used in NLP. The problem is that they are not always appropriate. In particular, they require the number of markables (segments that could be annotated) for their computation. If it is obvious for certain tasks like POS annotation, in which all the tokens are markables, it is less easy to determine in tasks in which the discrimination is not straightforward, like in gene renaming annotation.

To illustrate this, we introduce here the most widely used representation of data for inter-annotator agreement, the contingency table. This type of representation allows us not only to immediately visualize the agreement between annotators (the diagonal of the table), but also to rapidly identify the specifics of a campaign, like the prevalence of a category, i.e. the fact that a category is used (much) more often than the others. For these reasons, we strongly advocate for the presentation of the contingency table of an annotation campaign in the accompanying articles, whenever possible (two annotators and not too many categories), like in [PAL 05]. We completely agree with what is said in [HRI 02]:

> "showing the two-by-two contingency table with its marginal totals is probably as informative as any measure".

We present in Table 1.1 a contingency table for a toy POS annotation task with 5 categories and 100 segments, imagined from the following *Penn Treebank* example:

I/PRP do/VBP n't/RB feel/VB very/RB ferocious/JJ./.

In POS annotation, all the segments get an annotation, so there is no "hole" in the pavement. In this case, $A_o = 0.87$, $A_e^\kappa = 0.2058$, $A_e^\pi = 0.2062$, $\kappa = 0.8363$ and $\pi = 0.8362$.

		Annot. 1					
		PRP	VBP	RB	JJ	Punct	Total
Annot. 2	PRP	15	0	0	0	0	15
	VBP	2	17	1	2	0	22
	RB	0	2	22	3	0	27
	JJ	0	1	2	13	0	16
	Punct	0	0	0	0	20	20
	Total	17	20	25	18	20	100

Table 1.1. *(Imaginary) contingency table for a toy example of POS annotation*

On the contrary, in the gene renaming campaign, very few elements from the source are annotated and the empty category (no annotation) corresponding to the markables, is overwhelmingly prevalent, with 18,878 tokens (see Table 1.2).

		Annot. 1			
		Former	New	No annotation	Total
Annot. 2	Former	71	13	23	107
	New	8	69	15	92
	No annotation	7	8	18,840	18,855
	Total	86	90	18,878	19,054

Table 1.2. *Contingency table for the gene renaming annotation campaign [FOR 12c]*

Considering all the tokens as markables, we obtain $\kappa \approx \pi = 0.98$.

Obviously, we could have chosen to consider the gene names as markables instead of the tokens (see Table 1.3). In this case, we obtain $\kappa \approx \pi = 0.77$.

		Annot. 1			
		Former	New	No annotation	Total gene names
Annot. 2	Former	71	13	23	107
	New	8	69	15	92
	No annotation	7	8	951	966
	Total gene names	86	90	989	1,165

Table 1.3. *Contingency table for the gene renaming annotation campaign with the gene names as markables*

We detailed in [GRO 11] experiments that we led in the structured named entity annotation campaign on the inter-annotator agreement results in which we showed that the results vary quite significantly depending on the way the markables are computed.

The conclusion we draw from these various experiments is that coefficients from the kappa family should be avoided in cases in which there are "holes in the pavement", i.e. when not all of the signal is annotated, as in such cases, the necessarily arbitrary decisions in the definition of the markables may generate a prevalence bias.

1.4.2.4. *The F-measure: proceed with caution*

In some annotation campaigns, metrics usually used for the evaluation of the performance of the systems, like the F-measure, are used to evaluate the produced manual annotation. Often, this type of metric is chosen just because it is provided by default in the annotation tool, like in GATE (which also provides Cohen's kappa and Scott's pi). Sometimes, this choice is made to avoid the problem of the definition of the markables for the computation of kappa, for example in the case of named entity annotation [ALE 10, GRO 11]. In fact, it was demonstrated in [HRI 05] that when the number of markables is very high, the coefficients from the kappa family tend towards the F-measure.

The F-measure was designed for information retrieval and is now widely used in NLP. It corresponds to the weighted average of recall and precision:

$$\text{F-measure} = 2.\frac{\text{precision.recall}}{\text{precision} + \text{recall}}$$

with recall and precision defined as follows:

$$\text{Recall} = \frac{\text{Nb of correct found annotations}}{\text{Nb of correct expected annotations}}$$

$$\text{Precision} = \frac{\text{Nb of correct found annotations}}{\text{Total nb of annotations}}$$

It is therefore easy to compute.

By definition, precision and recall require a reference annotation. In the case of manual annotation, we are (most of the time) building this reference, so it does not exist yet. However, one may consider that the work of one annotator can be used as a reference for the other(s). The F-measure is then computed for each category and the global metric is the average of the local ones. It does not have to be computed both ways, as the recall of one annotator is the precision of the other [HRI 05].

However, the F-measure does not take chance into account, and we observed that sometimes chance has a significant impact on the results. This limitation makes it less suitable for manual annotation evaluation than other, more specific, measures like γ.

1.4.3. *Beyond kappas*

A lot of metrics have been proposed or revived, especially in the past few years, most of them to overcome the default of the kappa family metrics. We present here only a couple of them, from the weighted coefficients family, in order to introduce the final one, γ, which is very promising.

1.4.3.1. *Weighted coefficients*

Weighted coefficients allow us to give more importance to some disagreements than to others. The coefficients we briefly present here are more detailed in [ART 08]: the weighted version of Cohen's kappa (κ_ω) [COH 68] and Krippendorff's Alpha (α) [KRI 04].

Both coefficients are based on the disagreement between annotators and use a distance between categories, allowing us to describe how distinct two categories are. The idea behind this is that all disagreements are not equal, that some should have more weight than others. For example, a disagreement between two main categories (*Noun* and *Verb*), is more important than a disagreement in sub-types (*VerbPres* and *VerbPast*).

κ_ω and α are defined as follows:

$$\kappa_\omega, \alpha = 1 - \frac{D_0}{D_e}$$

where D_0 is the observed disagreement between the annotators and D_e the expected disagreement, i.e. the chance disagreement. The expected disagreements in κ_ω and α are computed in a similar way as κ and π respectively and include the notion of distance between categories.

We will not detail the calculus of D_e, which is presented for both metrics in [ART 08]. These metrics suffer from a major bias: distances are defined manually, based on intuition or knowledge of the campaign, and do not depend on the reality of the annotation. Another limitation is that they are dedicated to categorization tasks and do not take what Krippendorff calls unitizing into account.

Krippendorff then proposed a series of coefficients to go beyond α: αU [KRI 04], which covers unitizing, ${}_u\alpha$ [KRI 13], which focuses on positioning and ${}_{c|u}\alpha$ [KRI 13], which deals

with categories, but for now it has to be noted that ${}_u\alpha$ and ${}_{c|u}\alpha$ are not currently designed to cope with embedding or free overlapping between the units of the same annotator [MAT 15].

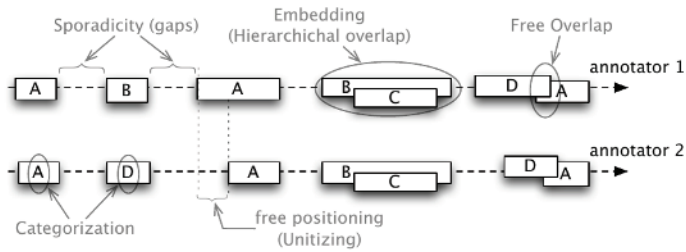


Figure 1.17. *Phenomena to take into account when computing inter-annotator agreements (Figure 1 from [MAT 15], by courtesy of the authors)*

1.4.3.2. γ: the (nearly) universal metrics

Yann Mathet and Antoine Widlöcher (GREYC-CNRS) designed the annotation tool `Glozz` and also created a new metric for the computation of the inter-annotator agreement, named γ,[28] which has been detailed in [MAT 15]. This metric is close to α in that it takes chance into account and does not require us to identify the markables. However, γ takes the nature of the units into account and, for example, two appended entities and a unique entity spanning two entities are considered differently in γ. The main advantage of γ is that it does not alter the annotations to compare them.

γ is holistic, in the sense that it takes the annotations from the whole corpus into account, rather than only local comparisons between units. It is also unified, as it does not dissociate between the alignment of the identified segments (discrimination and delimitation) and the agreement on the

28 An earlier version of this was called the "Glozz metrics" [MAT 11].

categories, both being performed simultaneously. Among the various possible alignments, γ keeps the one which minimizes the disagreement.

To do so, the metric relies on the notion of disorder of the system constituted by the set of annotations of a text. This disorder can be related to two types of dissimilarities, namely positional and categorial. The computation of the categorial dissimilarity requires, like in α and κ_ω, a distance between categories. This is the main weakness of the metric.

γ is defined as follows, for each annotation set j on a `corpus` c:

$$agreement(j) = \frac{e_{random}(c) - e(j)}{e_{random}(c)}$$

The entropy (the disorder) of an alignment of units corresponds to the average dissimilarities of its constituting units. The random entropy, $e_{random}(c)$, can be computed using different methods, including that presented in [MAT 11], which is implemented in `Glozz`. This method consists of observing the annotations produced by the annotators on the whole corpus and generating, independently of the text content, multi-annotations that respect the statistical distribution of the corpus, both in terms of positions and of categories.

The main issues with the metrics taking chance into account is that computing them is not always straightforward. It is especially the case for γ. The solution is to integrate the metric into an annotation tool, so that it can be computed directly using an interface. This is what has been done for γ, which is now rather easily computable in `Glozz`.

Interestingly, a technique is underused if it is not encapsulated in a tool, and if it is, it becomes a “black box” as

defined by Bruno Latour [LAT 87], i.e. something that is no longer open, therefore no longer questioned. This is exactly what happened with GATE and the F-measure. However, it is quite different with Glozz and γ, as the tool and the metric were created first for manual annotation and the metric was well-tested before being integrated into Glozz.

For the moment γ is still little used, so it does not really allow for a comparison with older annotation campaigns. We therefore suggest to compute it as a complement to kappa, whenever possible.

Contrary to the annotation tools, we think that with γ the domain is now close to a rather universal solution, even if using a distance that is determined *a priori* is constitutes an important bias. In addition, the inter-annotator agreement for relations is still a major open issue, with no appropriate solution in sight.

1.4.4. *Giving meaning to the metrics*

Ron Artstein and Massimo Poesio detail in [ART 08] the different scales of interpretation of the Kappas that were proposed over the years (see Figure 1.18) and emphasize the fact that it is very difficult to define a meaningful threshold. What is a "good" agreement, as measured by kappa or another metric?

They conclude with caution, proposing a threshold of "reasonable quality" of 0.8 for the kappas, while adding that they "doubt that a single cutoff point is appropriate for all purposes". Other works, in particular [GWE 12] that presents various inter-annotator agreement metrics, insist on the problem of their interpretation. Another related issue is how to compare two different results obtained with different metrics.

Some studies concerning the evaluation of the quality of manual annotation allowed us to identify factors influencing

the inter-annotator agreement, thus giving clues on the behavior of the metrics that were used. For example, it was demonstrated in [GUT 04] that the inter-annotator agreement and the complexity of the task are correlated (which is not surprising), in particular, the larger the tagset, the weaker the agreement. In the same article it is shown that there are only a limited number of categories generating disagreement. The meta study presented in [BAY 11] extends this research and identifies eight factors influencing the inter-annotator agreement: the "domain" (we would rather talk about the annotation type, as they compare word-sense disambiguation, prosodic transcriptions and phonetic transcriptions), the number of annotators, the training of the annotators, the annotation purpose, the knowledge of the domain, the language, the number of categories and the calculation method. The authors deduce from these recommendations to improve the quality of manual annotation. However, none of these analyses give a clear view on the behavior of agreement metrics or their meaning.

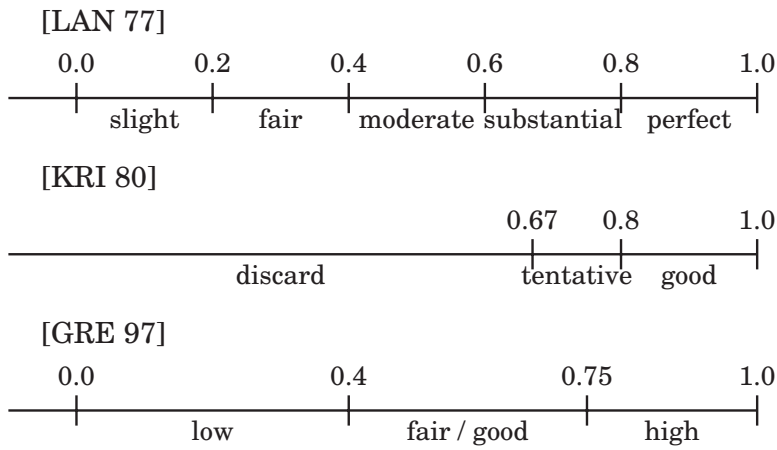


Figure 1.18. *Scales of interpretation of kappas (from the ESSLI 2009 course given by Gemma Boleda and Stefan Evert on inter-annotator agreement, by courtesy of the authors)*

We created a French working group on the subject, with people from LIMSI-CNRS (Sophie Rosset, Pierre Zweigenbaum, Cyril Grouin), LNE (Olivier Galibert and Juliette Kahn), INIST-CNRS (Claire François) and GREYC-CNRS (Yann Mathet and Antoine Widlöcher). Together, we discussed and reflected on the meaning of inter-annotator agreement metrics and the interpretation of the results. However, the original idea and the implementation of what is presented here come from Yann Mathet and Antoine Widlöcher. This work is detailed in [MAT 12]. We will present it here rapidly and complete it with real experiments led on the *TCOF-POS* corpus [BEN 12].

The idea proposed by Yann Mathet and Antoine Widlöcher is to reverse the problem and to analyze the results obtained with the various metrics on reference annotations (or artificial annotations), which are degraded in a controlled way.

1.4.4.1. *The Corpus Shuffling Tool*

This idea of applying controlled degradations to a reference is derived from research in thematic segmentation described in [PEV 02] and in [BES 09]. It was applied for the first time to inter-annotator agreement metrics in [MAT 11]. The principle is to generate degraded annotations in a statistically controlled way from a reference corpus. Several corpora are generated, corresponding to the different values of a deteriorated parameter, then the metrics are applied to the degraded corpora and their behavior can be observed.

The annotators produce errors that can be of various types and concern different dimensions. Each annotated unit can diverge from what it should be (a reference, imperfect by definition) in one or several ways:

– the delimitation of the unit is not correct (the frontiers do not correspond to the reference);

– the categorization of the unit is not correct (wrong category or wrong feature value);

– the discrimination of the unit is not correct: the annotation is not present in the reference (false positive);

– or, on the contrary, a unit from the reference is missing (false negative).

All these causes of errors in the annotation have to be taken into account in the inter-annotator agreement metrics. Mathet and Widlöcher developed a tool that generates "tremors" (i.e. degradations) along several dimensions.[29] These tremors are of various controlled magnitudes: the higher the magnitude, the more serious the errors. The obtained corpora with degraded annotations are then used to observe the behavior of the metrics according to different types of errors (dimensions) and a whole range of magnitudes. This allows us not only to compare the metrics (for a given magnitude, it is easy to compare the results obtained by the different metrics), but also to interpret the scores in a tangible manner (a given score for a given metric corresponds to a certain magnitude, of which we know the effects on the corpus). This tool takes as input the magnitude of error, from 0 (the perfect annotator) to 1 (the worst annotator, who annotates without even reading the text).

1.4.4.2. *Experimental results*

The experiments presented in [MAT 12] implied artificial annotations, i.e. annotations that were generated automatically from a statistical model describing the positional and categorical distribution of the markables. We will focus here on the obtained results rather than on the protocol, which is detailed in the article, and will present an experiment carried out on a real corpus.

29 This tool is freely available under a GPL license and will soon reappear here: http://www.glozz.org/corpusshufflingtool.

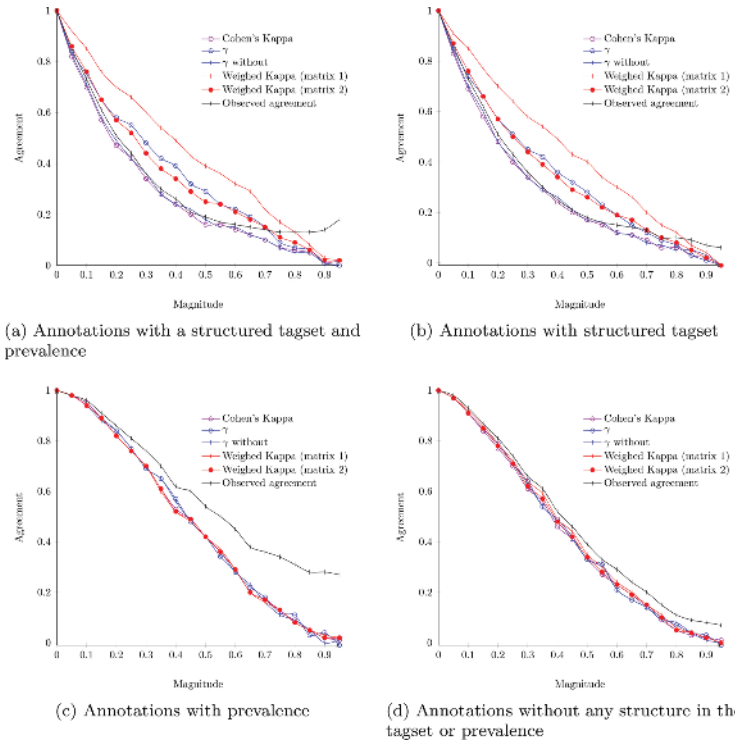
At the time of the experiments, the `Corpus Shuffling Tool` did not allow for a combination of paradigms to be taken into account. We therefore had to process segmentation and categorization separately. A first experiment was carried out on segmentation alone, rerunning the one described in [BES 09], comparing the generalized Hamming distance and WindowDiff, and adding γ. We will not present this experiment here, as it concerns metrics that cannot be considered as inter-annotator agreement metrics, since they require a reference. We simulated three annotators.

1.4.4.2.1. Artificial annotations

We present here results concerning the categorization process. The simulated situation is that of an annotation task in which the units to annotate are already localized, like in the *Penn Treebank* POS annotation. We created four annotation sets, including or not prevalence cases and a structured tagset, for which we consider that an error between a sub-category and a category should be considered as less serious than one between categories.

The `Corpus Shuffling Tool` was applied on these annotations to compare the following metrics: Cohen's kappa [COH 60], weighted kappa [COH 68], with two different weight matrices (the first one being much more lenient than the other) and γ, with or without the ability to deal with a structured tagset (taking the proximity between categories into account). An observed agreement (percentage of strict agreement between the three annotators) is also computed as baseline. The results of these experiments are shown in Figure 1.19.

These results show first that when there are no prevalence and no structured tagset (with different types of proximity between categories), all the compared metrics behave similarly (see Figure 1.19(d)), including the observed agreement (even if it slightly overestimates the agreement when the magnitude gets higher, because it does not take chance into account).



(a) Annotations with a structured tagset and prevalence

(b) Annotations with structured tagset

(c) Annotations with prevalence

(d) Annotations without any structure in the tagset or prevalence

Figure 1.19. *Comparison of the behaviors of the metrics on categorization. For a color version of the figure, see www.iste.co.uk/fort/nlp.zip*

In cases of prevalence of one category over the others (see Figure 1.19(c)), all the metrics continue to behave similarly, apart from the observed agreement, which tends to more and more overestimate the agreement, by nearly 0.25 at most. Chance has a significant impact here.

In case of a structured tagset, the weighted kappa and γ behave very differently than the other metrics. When taken into account, the more or less important proximity between categories, whether it is associated with prevalence or not (see Figures 1.19(a) and 1.19(b), respectively), generates

noticeable differences, of 0.15 for γ and 0.25 for the weighted kappa. This can easily be explained by the fact that these metrics use a matrix actually describing the proximity between categories. Moreover, it is interesting to note that when applying these two metrics to data without a structured tagset (or at least without taking it into account), they behave almost exactly the same way as the simpler metrics which do not take the proximity into account (bottom figures). These metrics (γ and weighted kappa) are not biased, whatever the corpus.

As for the observed agreement, it is closer to the other metrics in cases where the tagset is structured, probably due to the fact that in cases of average magnitudes, proximity is more influential than prevalence. However, with a magnitude above 0.6, the observed agreement overestimates the agreement again.

1.4.4.2.2. Annotations from a real corpus

Since we were limited by the fact that we could not combine various dimensions, we chose to focus on categories only. *TCOF-POS* [BEN 12] is a freely available corpus[30] of French spontaneous speech annotated in POS. It was perfect for our experiment, as the corpus was pre-segmented and the annotators did not have to review this, but only to categorize the units. In addition, this annotation task did not generate significant prevalence. However, the used tagset contained a hierarchy of types (*PRO:ind*, *PRO:dem*, *PRO:cls*, etc.), which has to be taken into account.

The results that we obtained are shown in Figure 1.20. They confirm the ones we got from artificial annotations. The observed agreement, which does not take chance into

30 The corpus is available here: http://www.cnrtl.fr/corpus/perceo/.

account, this time under-estimates the agreement. The weighted kappa seems to be the metrics that underestimates the least agreement in this case. This metric was computed from a user-defined matrix of weights, deduced from the annotation guide. These weights take into account the fact that an error between two categories (two types) is more serious than that between a category and its sub-categories. For example, the weight associated with an error between the following two sub-categories *Verb-PPRES* and *Verb-FUTUR* of the same category (*Verb*) is 0.5, whereas the weight associated with an error between two categories, like *Verb-PPRES* and *Noun* would be 1.

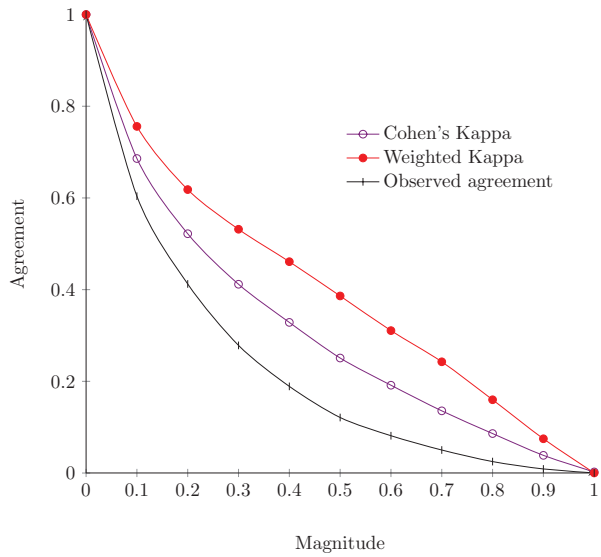


Figure 1.20. *Comparison of the behaviors of the metrics on categorization on the TCOF-POS corpus (no prevalence, but structure of the tagset taken into account). For a color version of the figure, see www.iste.co.uk/fort/nlp.zip*

Originally, the inter-annotator agreement on this corpus was computed using Cohen's kappa and reached 0.96. On the

Richter scale obtained using the `Corpus Shuffling Tool` and shown in Figure 1.20, this corresponds to a magnitude of 0.1, i.e. to a very limited deterioration. Therefore, we can say now without any doubt that the corpus is annotated in a very consistent way.

1.5. Conclusion

We covered here the bases for sound collaborative manual annotation: we detailled the annotation process, proposed a grid of analysis of the annotation complexity, gave an overview of the main annotation tools and exposed the potential biases in the annotation evaluation.

The next part of the book will be devoted to a major trend in NLP, crowdsourcing annotation. We will deconstruct the myths and show how reliable annotation can be obtain using ethical crowdsourcing.