Crowdsourcing and Aggregating Nested Markable Annotations

Chris Madge
Queen Mary University
c.j.madge@qmul.ac.uk

Juntao Yu
Queen Mary University
juntao.yu@qmul.ac.uk

Jon Chamberlain
University of Essex
jchamb@essex.ac.uk

Udo Kruschwitz
University of Essex
udo@essex.ac.uk

Silviu Paun
Queen Mary University
s.paun@qmul.ac.uk

Massimo Poesio
Queen Mary University
m.poesio@qmul.ac.uk

Abstract

One of the key steps in language resource creation is the identification of the text segments to be annotated, or **markables**– in our case, the (potentially nested) noun phrases in coreference resolution (or **mentions**). In this paper, we present a method for identifying markables for coreference annotation that combines high-performance automatic markable detectors with checking with a Game-With-A-Purpose (GWAP) and aggregation using a Bayesian annotation model. The method was evaluated both on news data and data from a variety of other genres and results in an improvement on F_1 of mention boundaries of over seven percentage points when compared with a state-of-the-art, domain-independent automatic mention detector, and almost three points over an in-domain mention detector. One of the key contributions of our proposal is its applicability to the case in which markables are **nested**, as is the case with coreference markables; but the GWAP and several of the proposed markable detectors are task- and language-independent and are thus applicable to a variety of other annotation scenarios.

1 Introduction

Developing Natural Language Processing (NLP) systems still requires large amounts of annotated text to train models, or as a gold standard to test the effectiveness of such models. The approach followed to create the most widely used data (Marcus et al., 1993; Palmer et al., 2005; Pradhan et al., 2012) is to separate the task of identifying the text segments to be annotated–the **markables**–from the annotation task proper. In our specific case, the markables of interest are the **mentions** used in coreference resolution, to be labelled as belonging to a coreference chain or as singletons; typical examples of mentions are pronouns, named entities, and other nominal phrases (Poesio et al., 2016). The annotation of mentions for coreference has similarities with the identification of the chunks for named entity resolution (NER), but mentions can and often are nested, as in the following example, from the *Phrase Detectives* corpus (Chamberlain et al., 2016)), where a mention of entity i is nested inside a mention of entity j.

[A wolf]$_i$ had been gorging on [an animal [he]$_i$ had killed]$_j$

The methods proposed in this paper are also applicable when markables are nested.

Mention identification for annotation is typically done semi-automatically, using first an automatic **mention detector** (or **extractor**) (Uryupina and Zanoli, 2016) and then checking its output by hand. Automatic mention detectors developed for coreference systems are generally used in the first step. Mention detection was recognized early on as a key step for overall coreference quality (Stoyanov et al., 2009; Hacioglu et al., 2005; Zhekova and Kübler, 2010; Uryupina and Zanoli, 2016), so a number of good quality mention detectors were developed, such as the mention detector included in the Stanford CORE pipeline (Manning et al., 2014), used by many of the top-performing systems in the 2012 CONLL Shared Task (Pradhan et al., 2012).[1] But this performance can be improved. The first contribution of this paper are new fully-trainable, language-independent mention detectors that outperform the Stanford CORE mention detector in a variety of genres.

But even the best automatic mention detectors do not achieve the accuracy required for high-quality corpus annotation, even when run in-

[1]Note that in many of the most recent systems mention detection is carried out as a joint inference task with coreference resolution (Peng et al., 2015)–e.g., by the current top performing system on the CONLL 2012 dataset, (Lee et al., 2018). These approaches generally result in better performance at coreference resolution, but not necessarily at mention detection. And even end-to-end systems require mention-annotated corpora for training and testing of course.

domain: the difference in performance between running coreference resolvers on gold mentions and running them on system mentions can be of up to 20 percentage points, and the results are even poorer when running such systems out-of-domain, for domains like biomedicine (Kim et al., 2011) or for under-resourced languages (Soraluze et al., 2012). So a manual checking step is still required to obtain high-quality results.[2]

Markable checking is increasingly done using **crowdsourcing** (Snow et al., 2008; Lawson et al., 2010; Bontcheva et al., 2017). But crowdsourcing, using microtask platforms such as Amazon Mechanical Turk can be too expensive for large scale annotation. For these cases, gamification tends to be a cheaper alternative (Poesio et al., 2013), also providing more accurate results and better contributor engagement (Lee et al., 2013).

The second contribution of this paper is an approach to mention detection for large-scale coreference annotation projects in which the output of mention detectors is corrected using a **Game-with-a-Purpose** (GWAP) (Von Ahn and Dabbish, 2008). A Game-With-A-Purpose is a game in which players label data as a by-effect of playing. GWAPs have been successful in many annotation projects (Lafourcade et al., 2015). Examples of successful GWAPs include *The ESP Game*, in which players contribute image labels (Von Ahn and Dabbish, 2004), and *FoldIt*, in which players solve protein-structure prediction problems (Cooper et al., 2010). However, so far there have not been any truly successful GWAPs for NLP. It has proven difficult to go from simple **gamification** of a labelling task to developing a proper game: e.g., in one of the best-known GWAPs for NLP, *Phrase Detectives* (Poesio et al., 2013), the labelling remains the core of the game dynamics. Yet, games such as *Puzzle Racer* have shown that engaging GWAPs producing annotations for text are possible. Furthermore, that the annotations thus collected are of a quality comparable to that obtainable using microtask crowdsourcing, and at a reduced cost (Jurgens and Navigli, 2014). However, such games have yet to achieve the player uptake or number of judgements comparable to GWAPs in other domains. Furthermore, it is not clear yet whether using GWAPs can result in better performance for tasks such as mention detection, for which good-performance systems exist.

In this work, automatically extracted mentions are checked using a two-player GWAP, *TileAttack*. Our previous analysis of the performance of *TileAttack* using player satisfaction metrics derived from the Free 2 Play literature suggests that we are succeeding in developing an engaging game (Madge et al., 2017). In this paper, we demonstrate that using *TileAttack* to check the output of our mention detector results in substantial improvement to the quality of the output. The game supports any text segmentation task, whether markables are nested or non-nested, aligned or not aligned, and is therefore applicable at least in principle to a variety of text annotation tasks besides coreference, including e.g., Named Entity Resolution (NER).

Key to this result is the use of a novel **aggregation method** to combine the labels produced by the mention detector with the labels collected using the game. A number of aggregation methods applicable to text segmentation labelling have been proposed (Dawid and Skene, 1979; Hovy et al., 2013; Passonneau and Carpenter, 2014; Felt et al., 2014; Rodrigues et al., 2014; Nguyen et al., 2017; Paun et al., 2018), but they are not directly applicable when markables can be nested. The third contribution of this paper is a novel method to use aggregation with potentially nested markables. We show that using this method to aggregate mention detector labels and *TileAttack* labels results in improved markable boundary quality.

2 Markables for coreference

Different coreference corpora adopt different definitions of markable (Poesio et al., 2016; Uryupina and Zanoli, 2016). The definition of (candidate) mention used in this paper is broadly speaking that adopted in corpora based on the MATE scheme (Poesio, 2004), such as ONTONOTES (Pradhan et al., 2012) ARRAU (Uryupina et al., 2019) and Phrase Detectives 1.0 (Chamberlain et al., 2016).

According to this definition, candidate mentions include all noun phrases (NPs) and all possessive pronouns. Non-referring NPs (like *It* in *It is sunny*

[2] One difference between the mention detectors used for coreference resolvers and those used to preprocess data for coreference annotation is relevant for subsequent discussion. The former usually aim for high recall and compromise on precision, placing more confidence/importance on the coreference resolution step (Kummerfeld et al., 2011) and being satisfied that incorrectly identified mentions will simply remain singletons which can be removed in post processing (Lee et al., 2011). The latter tend to go for high F. This difference played a role in our experiments, as discussed later.

or *a policeman* in *John is a policeman*) and singletons are considered candidate mentions as well, possibly to be filtered during coreference annotation proper.

The maximal projection of the NP is marked; i.e., the full extent of the NP including premodifiers, post-modifiers and appositions. In the case of a coordinated NP such as *Alice and Bob*, each conjunct and the coordinated NP are treated as candidate mentions:

[[Alice]$_i$ and [Bob]$_j$]$_k$ went to the shops. [They]$_k$ had a coffee.

3 Two automated mention detectors

The first ingredient of our proposal is two strong mention detectorsto serve both as baselines and as AI opponents for *TileAttack*.[3] The first pipeline first parses the input sentences using a dependency parser and then extracts mentions from the dependency parse; we call this the DEP pipeline. The second pipeline is a modified version of the neural named entity recognition system proposed by Lample et al. (2016); we call it NN pipeline. Both pipelines are trained on the Penn Treebank (PTB).

3.1 DEP pipeline

Our DEP pipeline first parses input sentences using the Mate dependency parser (Bohnet and Nivre, 2012), then applies a rule based mention extractor. Our extractor follows a three steps approach. It first extracts mention heads using heuristic patterns based on part-of-speech tags and dependency relations. The patterns are automatically extracted from the gold annotation of the Phrase Detectives 1.0 corpus (Chamberlain et al., 2016). We extract all the part-of-speech tags and dependency relations pairs of the mentions' head in the corpus, and use the most frequent patterns. The second step finds the maximum span related to a given mention head; for this we use the left/right-most direct or indirect children of the mention head as the start/end of the mention. The last step checks if any of the mentions created by step two overlap with each other. Overlapping mentions are replaced with the union of those mentions.

3.2 NN Pipeline(s)

Our second pipeline is based one the neural named entity recognition (NER) system proposed by Lample et al. (2016). This takes a sentence as the input and outputs a sequence of IOB style NER labels. The system uses a bidirectional LSTM to encode sentences and applies a sequential conditional random layer (CRF) over the output of the LSTM. But the CRF, effective when handling sequence labelling tasks such as NER, is not suitable for predicting mentions, as mentions can be nested. We address this as follows. For each token we create a maximum l candidate mentions. Let s, e be the start and end indices of the mention, and x_i the LSTM outputs on the i_{th} token. The mention is represented by $[x_s, x_e]$. We add a mention width feature embedding (ϕ) and apply a self-attention over the tokens inside a mention ($[x_s \ ... \ x_e]$) to create a weighted mention representation w_{se}. After creating the mention representation $[x_s, x_e, w_{se}, \phi]$, we use a feed-forward neural network with a sigmoid activation function on the output layer to assign each candidate mention a mention score. During training we minimise the sigmoid cross entropy loss. During prediction, mentions with a score above the threshold (t) are returned. The threshold can be adjusted to create models for different purposes. In particular, in this paper we experimented with two models: one optimized for high recall, the other for high F1. We use the same network parameters as Lample et al. (2016) except the two parameters introduced by our system. We set maximum mention width to 30 i.e. $l = 30$, and set $t = 0.5/0.95$ for our high-recall and high-F1 versions respectively.

Configuration	P	R	F_1
OntoNotes			
Stanford	40.38	89.46	55.65
DEP	36.60	83.79	50.95
NN High F1	73.53	74.01	73.77
NN High Recall	51.53	87.53	64.87
News			
Stanford	71.55	67.28	69.35
DEP	86.03	72.33	78.59
NN High F1	79.33	86.16	82.60
NN High Recall	71.65	91.29	80.29
Other Domains			
Stanford	77.52	80.11	78.79
DEP	84.72	81.78	83.22
NN High F1	79.92	87.48	83.53
NN High Recall	73.35	93.04	82.03

Table 1: Mention detectors comparison.

[3]The code is available at https://github.com/juntaoy/Dali-preprocessing-pipeline

3.3 Results

We use as a baseline the Stanford deterministic mention detector (Manning et al., 2014)–arguably, the most widely used mention detector for coreference with the CONLL dataset (Pradhan et al., 2012). Table 1 compares our pipelines and Stanford's on a number of data sets. Both of our pipelines consistently outperform the Stanford pipeline by a large margin.

4 *TileAttack*

To check the mentions produced by the automatic mention detectors discussed above we developed *TileAttack*, a web-based, two-player blind, token sequence-labelling game. Its visual design is inspired by *Scrabble*, with a tile-like visualisation shown in Figure 1. In the game, players perform a text segmentation task which involves marking spans of tokens represented by tiles. Players are awarded points based on player agreement.The game is highly adaptable to different corpora, sequence labelling tasks and languages.

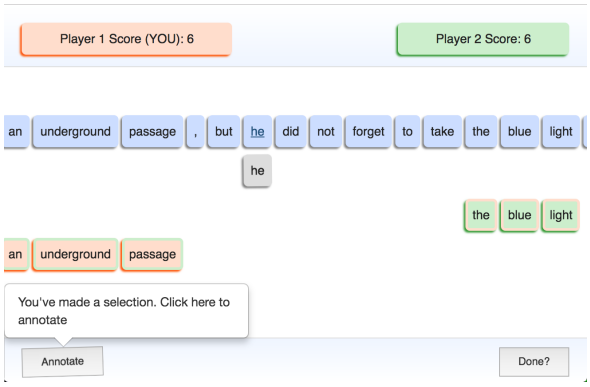

Figure 1: *TileAttack* screenshot

It is not easy to come up with an original game design. Our approach was to adopt a game design as close as possible to an existing recipe– specifically, the *ESP Game* (Von Ahn and Dabbish, 2008), but adapted to text annotation. Like *The ESP Game*, *TileAttack* has an "output-agreement" format, in which two players or agents are anonymously paired, and must produce the same output, for a given input (Von Ahn and Dabbish, 2008). This provided the opportunity to test what lessons learned from games similar to *The ESP Game* still apply to text annotation, games.

4.1 Gameplay

In each round, the player is shown a single sentence to annotate. Players can select a span from the sentence by simply selecting the start and end tokens of the item they wish to mark. A preview of their selection is then shown immediately below. To confirm this annotation, they may either click the preview selection or click the *Annotate* button. The annotation is then shown in the player's colour. When the two players match on a selection, the tiles for the selection in agreement are shown with a glinting effect, in the colour of the player that first annotated the tiles and a border colour of the player that agreed. The players' scores are shown at the top of the screen.

When players have finished, they click the *Done* button, upon which they will not be able to make any more moves, but will see their opponent's moves. Their opponent is also notified they have finished and invited to click *Done* once they have finished. Once both players have clicked *Done*, the round is finished and both players are shown a round summary screen. This screen shows the moves that both players agreed on, and whether they won or lost the round.

Clicking *Continue* then takes the player to a leaderboard showing wins, losses and the current top fifteen players. From this page they may click the *Next Game* button, to start another round. On the leaderboard, players are also offered the opportunity to sign up.

4.2 Opponents

Like all two-player GWAPs, *TileAttack* chooses an artificial agent as opponent for a player if no human opponent is available. An artificial agents is also used in crowdsourcing mode, as is the case with this study. The game uses three different artificial agents as opponents, selected in the following order of priority, descending to the next unless the condition is met:

Silver AI replays the aggregated result of all player games so far

Replay AI replays a recorded previous game - if a previous game is available for that item

Pipeline AI Plays the moves from the automated pipeline

5 Aggregating Mentions

The boundaries labelled by non-experts can be expected to be quite noisy compared to expert

annotations; but we can also expect the quality of the **aggregated** judgements to be comparable to that obtained with experts, provided sufficient non-experts are consulted (Snow et al., 2008). We are not aware however of any previous proposal to aggregate such annotations when they are nested. In this Section we introduce the two methods we used: a baseline on one based on taking the most popular judgement among the annotators (majority voting); and a probabilistic approach. Both these methods require a way for clustering together the mentions to be compared; we propose one such method in the first Section.

5.1 Head-based mention boundary clustering

To apply aggregation, it is necessary to determine which judgements (boundary pairs) are competing. We do this by clustering all annotations sharing the same **nominal head**.

The head of a player-generated candidate mention is extracted from the dependency parse used by the DEP pipeline after aligning the candidate mention with the dependency tree as follows. Given a player-generated candidate mention, we find first of all subtrees of the dependency tree that completely cover all the tokens in the candidate mention. The highest leftmost head of those subtrees is then considered as the head. If no nominal head is present in those subtrees, the candidate mention is not considered for aggregation.

Consider e.g. the sentence *John's car is red*. Suppose the players proposed the candidate mentions *John's car*, *John*, and the (incorrect) mention *John's car is*. Suppose also that the (automatically computed) dependency tree is as in Figure 2:

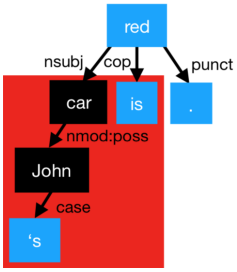

Figure 2: Finding a head for a proposed boundary

Then *John's car* can be aligned with the subtree whose head is *car*; *John's* can be aligned with a subtree with head *John*. Both of these heads are nominal, so the two candidate mentions are considered for clustering. *John's car is* would be aligned with the two subtrees with the roots *car* and *is*, shown in Figure 2 by the red box. The highest leftmost head and therefore the head that would be used is *car*. Relaxing the alignment criteria this way is important to allow the pipeline to guide the clustering while not constraining newly proposed boundaries to the pipeline's overall interpretation (which may be incorrect).

5.2 Baseline: Majority Voting

Majority Voting was used as a baseline aggregation method. Following clustering, majority voting is applied to each cluster, choosing the boundary that has the highest number of votes among all those sharing the same nominal head. Ties are broken randomly; the process is rerun five times.

5.3 A Probabilistic Approach

The majority vote baseline implicitly assumes equal expertise among annotators, an assumption shown to be false in practice (Dawid and Skene, 1979; Passonneau and Carpenter, 2014). A probabilistic model of annotation, on the other hand, can capture annotators' different levels of ability (Paun et al., 2018). This Section describes an application of the model proposed by (Dawid and Skene, 1979) to the boundary detection task.

Each of the clusters collected as discussed above contains a number of candidate boundaries supplied by the players. The goal is to identify the correct boundary for each cluster. A multi-class version of the Dawid&Skene model cannot be applied since the class space (the boundaries) is not consistent (i.e., the same set) across the clusters. However, a binary version of the model can be applied after some careful data pre-processing. Concretely, for each boundary we obtain a series of binary decisions as a result of a "one vs. the others" encoding performed at cluster-level. For example, given a cluster whose annotations are the boundaries "a, b, a, a", we have for the "a" boundary a collection of "1, 0, 1, 1" decisions, while for the "b" boundary we have "0, 1, 0, 0". We then train a Bayesian version of the binary Dawid&Skene model on these boundary decisions. The model infers for each boundary a decision indicator which can be interpreted as whether the boundary is correct or not. After some simple post-processing, we assign for each cluster the boundary whose posterior indicator has the most mass associated with the positive outcome.

6 Experimental Methodology

In order to evaluate our approach, we tested the mention boundaries obtained using the two proposed pipelines and by aggregating the judgements collected using *TileAttack* in several different ways over datasets in different genres.

6.1 Experiment setup

As said above, our approach to human checking of mentions produced by other players or by a system is to treat existing annotations as artificial agents that human players 'play against'. But we also pointed out that the mention detectors used for coreference resolution *systems* are optimised to achieve extremely high recall–the assumption being that the extra mentions will be filtered during coreference resolution proper–and that this optimisation may not be optimal when using an automatic mention detector for *annotation*–in our case, treating it as an agent from which the other players will derive feedback. In this context, a mention detector optimised for high overall F may be preferable, as it may provide better feedback to the human players. We tried therefore two versions of the NN pipeline in this experiment: one optimized for high recall, and one for high F_1. The two configurations are shown in Figure 3.[4]

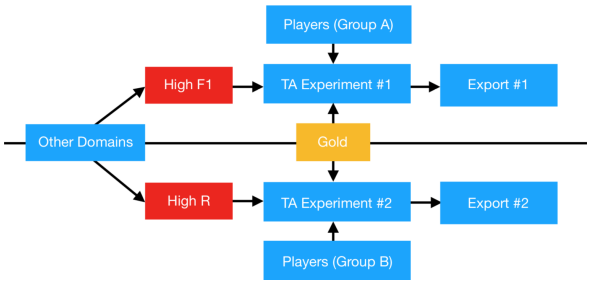

Figure 3: Experiment Setup

6.2 Participant recruitment and platform use

The regular players of *TileAttack* are typically experts in language or language puzzles, and many of them are linguists or computational linguists. As a result, the quality of the mentions they produce tends to be very high, as shown in Table 2, which reports the aggregated results of these players on the sentences from the 'Other Domains' dataset when playing against the 'High recall' pipeline. Our players obtain an aggregated F of 90.5, which is very high.

Aggregation Method	High Recall Pipeline		
	precision	recall	F_1
Majority Voting	90.284	87.536	88.889
Probabilistic	91.928	89.13	90.508

Table 2: Regular players accuracy on 'Other domains'

However, collecting judgements from real players tends to be slower than using a crowdsourcing service. Given that in this paper we were not concerned with comparing the effectiveness of crowdsourcing platforms and GWAPs, we collected the headline results for this experiment using judgements from participants recruited through Amazon Mechanical Turk. This was done for purely practical reasons–namely, ensuring we would collect sufficient data in a reasonably short time.

6.3 The participants' task

After completing the tutorial, participants are asked to annotate 3 sentences. At the end of each round, the participant is given feedback in the form of a comparison of their moves to those of the opponent. The participants are paid US $ 0.4 for completing the tutorial and three sentences on their first HIT, or five sentences on subsequent HITs.

6.4 Datasets

Most coreference datasets consist primarily of news text; for this reason, our first dataset, referred to below as "News", consists of 102 sentences from five randomly selected documents from the Wall Street Journal section of the Penn Treebank (Marcus et al., 1993), annotated with coreference as part of the ARRAU corpus (Uryupina et al., 2019).

The second dataset, referred to below as "Other Domains", is 180 sentences from a collection of our own creation consisting of documents covering different genres, from simple language learning texts and student reports, to Wikipedia pages and fiction from Project Gutenberg. We hand labelled the mentions in those sentences ourselves.

7 Results

7.1 News dataset

102 sentences were annotated by 131 participants. Each sentence was annotated at least 8 times (max-

[4]The DEP pipeline is not optimised either way.

imum of 11). A boundary was considered correct iff the start and end match *exactly*.

The results in Table 3 compare the results obtained using the four pipelines or application the two different aggregation approaches on the user (u), our DEP pipeline (d), NN (High F_1 and Recall configurations) and Stanford Pipeline (s). The presence or absence of the annotations for the users or pipelines is indicated by a preceding $+$ or $-$ respectively. MV indicates application of the majority voting aggregation method, and P the probabilistic aggregation method.

	Precision	Recall	F_1
Stanford	72.222	71.367	71.792
DEP	85.122	75.135	79.817
NN High F_1	78.090	83.151	80.541
NN High Recall	69.447	**88.833**	77.953
MV(+u -d -s)	80.293	70.786	75.240
MV(+u +d -s)	82.884	74.855	78.665
MV(+u +d +s)	77.542	78.794	78.163
MV(+u -d +s)	75.101	76.233	75.662
MV(+u +NN F_1)	85.578	77.706	81.452
MV(+u +NN R)	83.194	75.541	79.183
P(+u -d -s)	84.737	74.704	79.405
P(+u +d +s)	80.700	81.916	81.303
P(+u +d -s)	**86.770**	78.364	**82.353**
P(+u -d +s)	78.025	79.117	78.568
P(+u +NN F_1)	86.587	78.247	82.206
P(+u +NN R)	85.697	77.814	81.566

Table 3: Comparing pipeline and aggregation methods

The Table suggests, first of all, that the domain-trained pipelines outperform the domain-independent Stanford one, as expected. Second, that in this genre human judgements only match the domain-dependent pipelines when probabilistic aggregation is used. Third, that by aggregating user judgements and domain-dependent pipelines we see an improvement in F1 of up to 2.5 percentage points, but only with probabilistic aggregation.

In Figures 4 and 5 we plot F_1 to look at how many non-expert annotators are required to rival the performance of the pipelines using the respective aggregation methods. In Figure 4 only the participants are shown. The Figure shows that in this genre the domain-specific automated pipeline (trained on this domain) outperforms the participants, but already at five annotators, aggregated with the probabilistic aggregation method, we are very close to the performance of the domain specific pipeline. And in Figure 5, which shows the results aggregating participants with the pipelines (and in which the first two participants are the two automated pipelines), we can see that we only need to aggregate 3 participants to the domain-specific pipeline to exceed its performance.

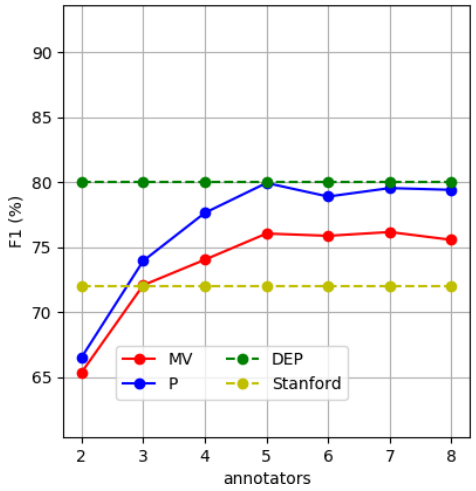

Figure 4: Human annotators F_1

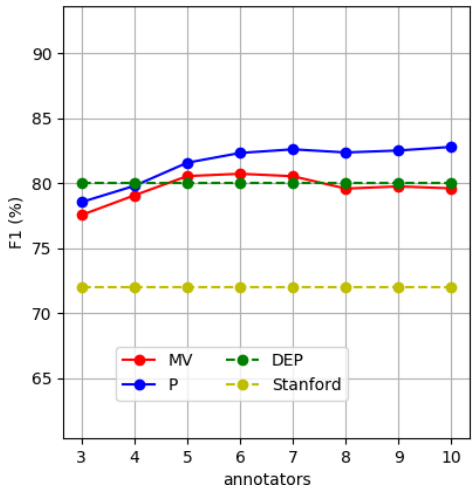

Figure 5: Aggregated users and pipelines (first two annotators are automated pipelines) F_1

7.2 Other Domains

431 participants in the High Recall Group and 120 participants in the High F_1 Group labelled 180 sentences.

Table 4 shows the results for both configurations of the pipeline (highest score marked in bold). We can see that operating out of their original domains, the automated pipelines, while still outperforming the Stanford pipeline by around 4 percentage points, do not outperform aggregated users.

However, they do appear to serve well as agents to train participants to perform annotations, as participants annotate to a high level of accuracy.[5]

7.3 Error Analysis

We analysed the errors produced both before and after aggregation. There were many errors to consider, so we took an approximate rule driven approach to characterise as many as possible.

Before aggregation, by far the most common error (1254 cases) is participants marking individual nouns as noun phrases (e.g., marking *the [cat]* instead of *[the cat]*). This suggests a misunderstanding of what a noun phrase is that may possibly be addressed by improvements to the tutorial. Similarly, in 606 cases participants mark named entities/strings of proper nouns rather than the encapsulating noun phrase.

The next most common error (529 cases) is annotators neglecting to include post-modifiers when selecting noun phrase boundaries (e.g., marking *[the cat] in the hat* instead of *[the cat in the hat]*. This is often the most popular judgement, and as such, chosen by MV. A real example of this is in Figure 6: whilst five annotators did identify the correct boundaries (in green), matching the gold standard (in gold), more (six), only marked the reduced boundaries (in red) "A consortium of private investors". This sequence, missing the post-modifier, was consequently chosen by majority voting. The probabilistic method (in silver), however, expressed more confidence in the five annotators and provided a correct final judgement.

Count		0	1	2	3	4	5	6	7	8	9	10
		A	consortium	of	private	investors	operating	as	LJH	Funding	Co.	said
gold	0, 9	A	consortium	of	private	investors	operating	as	LJH	Funding	Co.	
silver	0, 9	A	consortium	of	private	investors	operating	as	LJH	Funding	Co.	
6	0, 4	A	consortium	of	private	investors						
5	0, 1	A	consortium									
5	0, 9	A	consortium	of	private	investors	operating	as	LJH	Funding	Co.	

Figure 6: Example of post-modifier phrase

In the texts from "Other Domains", one of most common errors produced by the automated pipelines in in cases of coordination, as in

> Sammy chose ten [books and the library] said he could borrow them for one month.

where "ten books" and "the library" should not be coordinated.

[5] As pointed out, workers do not do as well as players recruited to *TileAttack* by more organic means (Table 2).

Another common problem for automated mention detectors was prepositional phrase attachment. Our automated mention detectors tend to prefer low attachment, as in

> So John and Caroline filled up a [green bin with mandarins].

The example above highlights another common error with the mention detectors, missing the determiner - most commonly, quantifiers and indefinite articles.

Lastly, proper nouns near the start of sentences are often incorrectly grouped with the capitalized first token which is incorrectly also identified as a proper noun (e.g. *[First Art] sat in the car...* rather than *First [Art] sat in the car...*)

8 Related Work

8.1 Gamifying all steps of a pipeline

The Groningen Meaning Bank project includes multiple gamified interfaces as part of a platform called *Wordrobe*. These gamified interfaces are supported by prior judgements provided by an automated NLP pipeline and the *GMB Explorer* (Basile et al., 2012).

The *Wordrobe* suite of games (Bos et al., 2017) includes multiple games that go on to produce similar annotations to that of *TileAttack* (e.g. Named Entity Recognition). However, all tasks are represented by a single common multiple choice format. targets a single yet core NLP annotation task (sequence labelling) with a broad set of applications. We do not constrain user input based on any prior judgement beyond tokenisation.

8.2 Aggregating markable annotations

Whilst there has been a great deal of work and evaluation on aggregating judgements from noisy crowdsourced data, this is generally focused on classification-based annotations (Sheshadri and Lease, 2013) and does not generalise to sequence labelling tasks like NER markable annotation. Dredze et al. proposed both a "Multi-CRF" approach to aggregating noisy sequence labels, and including judgements provided by an automated pipeline, in a NER task (Dredze et al., 2009). Confidence in annotators is not modelled. However, it has been extended to incorporate the reliability of the annotator with a similar method that also combines Expectation Maximization with CRF in an NER and NP chunking task (Rodrigues et al., 2014). Nguyen et al. apply HMM and LSTM methods to aggregating judgements in NER and

	High Recall Experiment			High F1 Experiment		
	precision	recall	F_1	precision	recall	F_1
Stanford	77.524	80.111	78.796	77.524	80.111	78.796
MV(users Stanford)	82.152	87.065	84.537	82.260	87.065	84.595
P(users Stanford)	82.438	87.483	84.885	82.523	87.344	84.865
DEP	84.726	81.780	83.227	84.726	81.780	83.227
MV(users DEP)	**88.434**	87.204	**87.815**	**87.729**	86.509	**87.115**
P(users DEP)	87.870	86.648	87.255	87.588	86.37	86.975
NN	73.355	**93.046**	82.036	79.924	87.483	83.533
MV(users NN)	81.472	89.291	85.202	80.000	89.013	84.266
P(users NN)	81.807	89.430	85.449	84.363	**89.291**	86.757
MV(users)	87.977	85.349	86.643	86.533	84.006	85.251
P(users)	88.270	85.633	86.931	82.523	87.344	84.865

Table 4: Results on the 'Other Domains' dataset (rounded to 3 dp)

IE, including a crowd component in both models representing each annotators ability for each label class (Nguyen et al., 2017).

Whilst variations of CRF and HMM have demonstrated a great improvement over majority voting approaches, models to date have not taken into account the nested nature of sequences that occur in tasks such as markable identification.

9 Discussion and Conclusions

In this paper, we presented a hybrid mention detection method combining state-of-the-art automatic mention detectors with a gamified, two-player interface to collect markable judgements. The integration takes place by using the automatic mention detectors as 'players' in the game. Data from automatic mention detectors and players are then aggregated using a probabilistic aggregation method choosing the most likely interpretation in a nominal head-centered cluster.

We showed that using this combination we can achieve, in the news domain, an accuracy at mention identification that is almost three percentage points higher than that obtained with an automatic domain-trained mention detector, and over seven percentage points higher than that obtained with a domain-independent one. We also test the approach in genres outside those in which the automatic pipelines were trained, showing that high accuracy can be achieved in these as well. These results suggest that it may be possible to gamify not just the task of annotating coreference, but also the prerequisite steps to that.

As a rule of thumb, of the two best-known forms of crowdsourcing, microtask crowdsourcing using platforms such as Amazon Mechanical Turk is best to label small to medium size amounts of data in a short time, and for labelling data of no intrinsic interest. Whereas crowdsourcing with games-with-a-purpose is best in cases when the objective is to collect very large amounts of labels, so that the initial costs for setting up the game can be offset by the reduced costs of labelling (Poesio et al., 2013). One example in point is the *Phrase Detectives* annotation effort. The latest release of these data (Poesio et al., 2019) contains 2.2M judgments, around 4 times the number of judgments collected for ONTONOTES. The approach to mention detection proposed in this paper was developed in support of games such as *Phrase Detectives*, thus a GWAP or at least gamified approach as exemplified by *TileAttack* was deemed more appropriate even if the judgments used in this paper were collected using Amazon Mechanical Turk for speed. About 5,000 sentences were annotated by regular (i.e., not paid) players in this initial development phase, but we expect the game will be able to collect a comparable amount of judgments as for *Phrase Detectives* once it's fully operational and properly advertised. Anda gamified interface such as *TileAttack* can be beneficial even for projects who just use microtask crowdsourcing, as it has been shown that gamified HITs are more popular (Morschheuser et al., 2017).

Acknowledgements

This research was supported in part by the EPSRC CDT in Intelligent Games and Game Intelligence (IGGI), EP/L015846/1; in part by the DALI project, ERC Grant 695662.

References

Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. 2012. Developing a large semantically annotated corpus. In *LREC 2012, Eighth International Conference on Language Resources and Evaluation*.

Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1455–1465, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kalina Bontcheva, Leon Derczynski, and Ian Roberts. 2017. Crowdsourcing named entity recognition and entity linking corpora. In *Handbook of Linguistic Annotation*, pages 875–892. Springer.

Johan Bos, Valerio Basile, Kilian Evang, Noortje Venhuizen, and Johannes Bjerva. 2017. The Groningen Meaning Bank. In Nancy Ide and James Pustejovsky, editors, *Handbook of Linguistic Annotation*, volume 2, pages 463–496. Springer.

Jon Chamberlain, Massimo Poesio, and Udo Kruschwitz. 2016. Phrase detectives corpus 1.0: Crowdsourced anaphoric coreference. In *Proc. of LREC*, Portoroz, Slovenia.

Seth Cooper, Firas Khatib, Adrien Treuille, Janos Barbero, Jeehyung Lee, Michael Beenen, Andrew Leaver-Fay, David Baker, Zoran Popović, et al. 2010. Predicting protein structures with a multiplayer online game. *Nature*, 466(7307):756–760.

A. Philip Dawid and Allan M. Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied Statistics*, 28(1):20–28.

Mark Dredze, Partha Pratim Talukdar, and Koby Crammer. 2009. Sequence learning from data with multiple labels. page 39.

Paul Felt, Robbie Haertel, Eric K. Ringger, and Kevin D. Seppi. 2014. MOMRESP: A bayesian model for multi-annotator document labeling. In *Proc. of LREC*, Reykjavik.

Kadri Hacioglu, Benjamin Douglas, and Ying Chen. 2005. Detection of entity mentions occurring in english and chinese text. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 379–386. Association for Computational Linguistics.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with MACE. In *Proc. of NAACL*, pages 1120–1130.

David Jurgens and Roberto Navigli. 2014. It's all fun and games until someone annotates: Video games with a purpose for linguistic annotation. *TACL*, 2:449–464.

Youngjun Kim, Ellen Riloff, and Nathan Gilbert. 2011. The taming of reconcile as a biomedical coreference resolver. In *Proceedings of the BioNLP Shared Task 2011 Workshop*, pages 89–93. Association for Computational Linguistics.

Jonathan K Kummerfeld, Mohit Bansal, David Burkett, and Dan Klein. 2011. Mention detection: heuristics for the ontonotes annotations. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 102–106. Association for Computational Linguistics.

Mathieu Lafourcade, Alain Joubert, and Nathalie Le Brun. 2015. *Games with a Purpose (GWAPS)*. John Wiley & Sons.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270. Association for Computational Linguistics.

Nolan Lawson, Kevin Eustice, Mike Perkowitz, and Meliha Yetisgen-Yildiz. 2010. Annotating large email datasets for named entity recognition with mechanical turk. In *Proceedings of the NAACL HLT 2010 workshop on creating speech and language data with Amazon's Mechanical Turk*, pages 71–79. Association for Computational Linguistics.

Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2011. Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 28–34. Association for Computational Linguistics.

Kenton Lee, Luheng He, and Luke S. Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proc. of NAACL*.

Tak Yeon Lee, Casey Dugan, Werner Geyer, Tristan Ratchford, Jamie C Rasmussen, N Sadat Shami, and Stela Lupushor. 2013. Experiments on motivational feedback for crowdsourced workers. In *ICWSM*.

Chris Madge, Jon Chamberlain, Udo Kruschwitz, and Massimo Poesio. 2017. Experiment-driven development of a gwap for marking segments in text. In *Extended Abstracts Publication of the Annual Symposium on Computer-Human Interaction in Play*, CHI PLAY '17 Extended Abstracts, pages 397–404, New York, NY, USA. ACM.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proc. of the 52nd ACL (system demos)*, pages 55–60.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Benedikt Morschheuser, Juho Hamari, Jonna Koivisto, and Alexander Maedche. 2017. Gamified crowdsourcing: Conceptualization, literature review, and future agenda. *International Journal of Human-Computer Studies*, 106:26–43.

An T Nguyen, Byron C Wallace, Junyi Jessy Li, Ani Nenkova, and Matthew Lease. 2017. Aggregating and predicting sequence labels from crowd annotations. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2017, page 299. NIH Public Access.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

Rebecca J. Passonneau and Bob Carpenter. 2014. The benefits of a model of annotation. *Transactions of the ACL*, 2:311–326.

Silviu Paun, Bob Carpenter, Jon Chamberlain, Dirk Hovy, Udo Kruschwitz, and Massimo Poesio. 2018. Comparing Bayesian Models of Annotation. *Transactions of the Association for Computational Linguistics*, 6:571–585.

Haoruo Peng, Kai-Wei Chang, and Dan Roth. 2015. A joint framework for coreference resolution and mention head detection. In *CoNLL*, volume 51, page 12.

Massimo Poesio. 2004. The MATE/GNOME scheme for anaphoric annotation, revisited. In *Proc. of SIGDIAL*, Boston.

Massimo Poesio, Jon Chamberlain, Udo Kruschwitz, Silviu Paun, Alexandra Uma, and Juntao Yu. 2019. A crowdsourced corpus of multiple judgments and disagreement on anaphoric interpretation. In *Proc. of NAACL*, Minneapolis. Association for Computational Linguistics.

Massimo Poesio, Jon Chamberlain, Udo Kruschwitz, Livio Robaldo, and Luca Ducceschi. 2013. Phrase detectives: Utilizing collective intelligence for internet-scale language resource creation. *ACM TiiS*, 3(1):3.

Massimo Poesio, Sameer Pradhan, Marta Recasens, Kepa Rodriguez, and Yannick Versley. 2016. Annotated corpora and annotation tools. In M. Poesio, R. Stuckardt, and Y. Versley, editors, *Anaphora Resolution: Algorithms, Resources and Applications*, chapter 4. Springer.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Proceedings of the Sixteenth Conference on Computational Natural Language Learning (CoNLL 2012)*, Jeju, Korea.

Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. 2014. Sequence labeling with multiple annotators. *Machine learning*, 95(2):165–181.

Aashish Sheshadri and Matthew Lease. 2013. Square: A benchmark for research on computing crowd consensus. In *First AAAI Conference on Human Computation and Crowdsourcing*.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y Ng. 2008. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 254–263. Association for Computational Linguistics.

Ander Soraluze, Olatz Arregi, Xabier Arregi, Klara Ceberio, and Arantza Díaz De Ilarraza. 2012. Mention detection: First steps in the development of a basque coreference resolution system. In *KONVENS*, pages 128–136.

Veselin Stoyanov, Nathan Gilbert, Claire Cardie, and Ellen Riloff. 2009. Conundrums in noun phrase coreference resolution: Making sense of the state-of-the-art. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 656–664. Association for Computational Linguistics.

Olga Uryupina, Ron Artstein, Antonella Bristot, Federica Cavicchio, Francesca Delogu, Kepa Rodriguez, and Massimo Poesio. 2019. Annotating a broad range of anaphoric phenomena, in a variety of genres: the ARRAU corpus. *Journal of Natural Language Engineering*.

Olga Uryupina and Roberto Zanoli. 2016. Preprocessing. In M. Poesio, R. Stuckardt, and Y. Versley, editors, *Anaphora Resolution: Algorithms, Resources and Applications*, chapter 6, pages 209–236. Springer.

Luis Von Ahn and Laura Dabbish. 2004. Labeling images with a computer game. In *SIGCHI*, pages 319–326. ACM.

Luis Von Ahn and Laura Dabbish. 2008. Designing games with a purpose. *Communications of the ACM*, 51(8):58–67.

Desislava Zhekova and Sandra Kübler. 2010. Ubiu: A language-independent system for coreference resolution. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 96–99. Association for Computational Linguistics.