HyEnA: A Hybrid Method for Extracting Arguments from Opinions

Michiel VAN DER MEER [a,1], Enrico LISCIO [b], Catholijn M. JONKER [a,b],
Aske PLAAT [a], Piek VOSSEN [c] and Pradeep K. MURUKANNAIAH [b]
[a] *Leiden Institute for Advanced Computer Science (LIACS), Leiden University*
[b] *Interactive Intelligence (II), TU Delft*
[c] *Computational Lexicology and Terminology Lab (CLTL), VU*

Abstract. The key arguments underlying a large and noisy set of opinions help understand the opinions quickly and accurately. Fully automated methods can extract arguments but (1) require large labeled datasets and (2) work well for known viewpoints, but not for novel points of view. We propose HyEnA, a hybrid (human + AI) method for extracting arguments from opinionated texts, combining the speed of automated processing with the understanding and reasoning capabilities of humans. We evaluate HyEnA on three feedback corpora. We find that, on the one hand, HyEnA achieves higher coverage and precision than a state-of-the-art automated method, when compared on a common set of diverse opinions, justifying the need for human insight. On the other hand, HyEnA requires less human effort and does not compromise quality compared to (fully manual) expert analysis, demonstrating the benefit of combining human and machine intelligence.

Keywords. hybrid intelligence, argument extraction, natural language processing

1. Introduction

To make decisions on large public issues, such as combating the COVID-19 pandemic and transitioning to green energy, policy makers often turn to the citizens for feedback [1,2]. This feedback provides insights into public opinion and contains diverse perspectives. Further, involving the public in the decision-making process helps in gaining their support when the decisions are to be implemented.

In the face of crises, decisions must be made swiftly. Thus, the collection of feedback, its analysis, and recommendations for decision-making are made under tight time constraints. For example, when debating on relaxing COVID-19 measures in the Netherlands, researchers had one month to design the experiment, collect public feedback, and make recommendations [3]. The time constraint limits the amount of information researchers can look at, potentially painting an incomplete picture of the opinions. In the scenario above, researchers analyzed data manually and thus could analyze less than 8% of the qualitative feedback provided by more than 25,000 participants.

[1] Corresponding Author: Michiel van der Meer, Leiden Institute for Advanced Computer Science, Niels Bohrweg 1, 2333 CA Leiden, The Netherlands; E-mail: m.t.van.der.meer@liacs.leidenuniv.nl.

Argument Mining (AM) [4] methods can assist in increasing the efficiency of feedback analysis by, e.g., separating strongly argumentative feedback from noise and classifying statements as supporting or opposing a decision. However, applying AM methods for feedback analysis poses three main challenges. First, AM methods generalize poorly across domains [5,6]. Thus, they require large amounts of domain-specific training data, which is often not available. While contextualized representations, using the pre- or fine-tuning paradigm, yield more promising results [7,8], they, too, rely on large amounts of data. Second, although AM methods can detect logical connections between comments and policy decisions, they do not compress the information. That is, they do not recognize whether two identified arguments describe the same concept, leaving the policy makers with significant manual labor. Finally, analyzing a small sample of comments might cause minority opinions to be ignored [9], creating a bias toward popular (repeated) arguments, which can perpetuate echo chambers and filter bubbles [10,11].

The *key point analysis* (KPA) task [12] seeks to automatically compress argumentative discourse into unique *key points*, which can be matched to arguments. However, synthesizing key points is a significant challenge. In the ArgKP dataset, domain experts (skilled debaters) were asked to generate key points. Subsequently, a model was trained to take over the task [13]. However, such key points are not grounded in data (public opinion) and are subject to the perspectives and biases of the human experts. Further, making use of a few experts to generate key points defeats the purpose of engaging the larger public in the decision-making process.

We argue for a joint human-machine approach, exploiting both the speed of automated methods and the human understanding of subtle issues. We propose HyEnA (Hybrid Extraction of Arguments), a hybrid (human + AI) method for extracting a diverse set of key arguments from a textual opinion corpus. HyEnA breaks down the argument extraction task into argument *annotation* and *consolidation* phases. In each phase, HyEnA employs human (crowd) annotators and supports them via intelligent algorithms based on natural language processing (NLP) as shown in Figure 1.

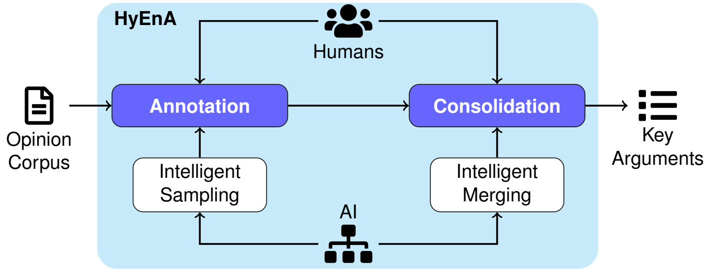

Figure 1. Overview of the HyEnA method.

We evaluate our method on three corpora, each containing more than 10K public opinions on relaxing COVID-19 restrictions [3]. We compare HyEnA with an automated approach [13] performing the KPA task. In addition, we compare the key arguments generated by HyEnA with manually obtained insights identified by experts [3].

Contributions (1) We present a hybrid method for key argument extraction, which, given a collection of opinionated user comments, generates a diverse set of key arguments raised in the discussion. (2) We evaluate our method on real corpora of public feedback on policy options. Compared to an automated baseline, HyEnA increases the precision of the key arguments produced and improves coverage over diverse opinions. Compared to the manual baseline, HyEnA identifies a large portion of arguments identified by experts as well as new arguments that experts did not identify.

2. Related work

We describe related works on AM and methods for extracting key arguments.

2.1. Computational Argument Analysis

Argument Mining methods [14,4] focus on computational analysis of arguments. They seek to discover arguments brought forward by speakers and identify connections between them. AM is a costly and complex process, and it often requires significant effort by human annotators for reaching moderate inter-rater agreement [15]. The ability to recognize and extract arguments from text is dependent on the argumentativeness of the underlying data. Given argumentative texts, popular NLP models are reasonably good at recognizing argumentative discourse [16,17,7]. Typically, the first step of AM is to identify the elemental components of arguments (e.g., *claims* and *premises*) in text [18]. The combination of such components forms a structured argument. However, there is currently no consensus on the exact linguistic notion of such elemental components [19]. Nonetheless, a few characteristics have been recognized as important for recognizing arguments, namely that arguments (1) contain (informal) logical reasoning [20], (2) address a *why* question [21], and (3) have a non-neutral stance towards the issue being discussed [20].

HyEnA is a novel AM method that combines human annotators and automated NLP models. By splitting up the argument extraction task into distinct phases, we take advantage of the diverse human perspectives, while addressing scalability problems through automation. Because annotators are only given the opinion text, we aim to achieve better grounding by preserving links between the argument and the original text, all while providing condensed key arguments useful in analysis.

2.2. Summarization of Arguments

Automated methods have been proposed to create a core set of key points from a large corpus of individual comments [13]. In this paradigm, comments are filtered by a manually tuned selection heuristic, resulting in a list of key point candidates. The candidates are matched against all comments, based on a classifier trained for the argument–key point matching task [12]. We evaluate the performance of this approach on a novel domain on COVID-19 measures and compare it against HyEnA.

Additionally, there exists a body of work on Natural Language Inference (NLI) and Semantic Textual Similarity (STS). In these works, models are trained to indicate semantic similarity or logical entailment between two sentences [22,23]. They have made a significant impact on general-purpose applications [24,25]. However, downstream applications often need additional fine-tuning [26] in order to perform a task well. They also capture generic aspects of semantic similarity and entailment, which may not be applicable to arguments [23], or conversely overfit to spurious patterns in the data [27].

3. Method

HyEnA is a hybrid method since it combines automated techniques and human judgement [28]. HyEnA guides human annotators toward the creation of *key arguments* (i.e., semantically distinct arguments that describe relevant aspects of the topic under discussion) from an *opinion corpus* composed of individual *opinions* (i.e., textual comments) on the topic of discussion.

HyEnA consists of two phases (Figure 1). In the first phase (*Key Argument Annotation*), an intelligent sampling algorithm guides human annotators through an opinion corpus to extract high-level information from the opinions. In the second phase (*Key Argument Consolidation*), a new group of annotators merges the results from the first phase, supported by an intelligent merging strategy, involving manual and automatic labeling. In the second phase, HyEnA aims to reduce the subjectivity in the annotation. The final result of HyEnA is key arguments grounded on the opinions in the corpus.

3.1. Opinion Corpora

Our opinion corpora are composed of citizens' feedback on COVID-19 relaxation measures, a contemporary topic. The feedback was gathered in April and May 2020 using the Participatory Value Evaluation (PVE) method [3]. In the PVE, participants are offered a set of policy options and asked to select their preferred portfolio of choices. Then, the participants are asked to motivate why they picked certain options (*pro* stance) and not pick the other options (*con* stance) via textual comments. Pro- and con-opinions together form the opinion corpus. We analyze feedback from 26,293 Dutch citizens on three of these options, treating comments on each option as an opinion corpus. Table 1 shows examples. In our experiments, the HyEnA method is applied to one corpus at a time. For each policy option, we use the keyword in uppercase as the *option* identifier in the remainder of the paper. The opinions in these corpora are similar to noisy user-generated web comments [29]. Some opinions span multiple sentences and contain more than one argument.

Table 1. Example opinions in the COVID-19 corpora.

Policy option (Corpus)	**Example opinion**	**# Opinions**
YOUNG people may come together in small groups	Then they can go back to school (Pro)	13400
All restrictions are lifted for persons who are IMMUNE	Encourages inequality (Con)	10567
REOPEN hospitality and entertainment industry	The economic damage is too high (Pro)	12814

The original opinions were provided in Dutch. To accommodate a diverse set of annotators in our experiments, we translated all comments to English using the Microsoft Azure Translation service. All experiments are performed with the translated opinions. Mixing (pretrained) embeddings and machine-translated comments has a minimal impact on downstream task performance [30,31,32]. Although all experiments are conducted in English, the link to the original Dutch text is preserved for future applications.

3.2. Key Argument Annotation

In the first phase of HyEnA, human annotators extract individual key argument lists by analyzing the opinion corpus. Since a realistic corpus consists of thousands of opinions, it is unfeasible for an annotator to read all opinions. Thus, HyEnA proposes a fixed number of opinions to each annotator. HyEnA employs NLP and a sampling technique to select diverse opinions to present to an annotator.

Intelligent Opinion Sampling Each annotator is presented, one at a time, with a fixed number of opinions. To sample the next opinion, we embed all opinions and arguments observed thus far using the S-BERT model (M_S) [23]. S-BERT converts sentences into fixed-length embeddings, which can be used to compute semantic similarities between pairs of sentences.

Then, we select a pool of candidate opinions using the Farthest-First Traversal (FFT) algorithm [33]. FFT selects the candidate pool as the f farthest opinions in the embedding space from the previously read opinions and annotated arguments (in our experiments, we empirically select $f = 5$). Next, we use an argument quality classifier trained on the ArgQ dataset [34] to select the opinion most clear and related to the policy option. In this way, we aim at increasing both the diversity and quality of the opinions presented to each annotator.

Annotation Upon reading an opinion, the annotator is asked, first, to *identify* whether the opinion contains an argument or not. If so, the annotator is asked to check whether the argument is already included in their current list of key arguments. If it is not, the annotator should *extract* the argument into a standalone expression (i.e., into a key argument), and add it to the list of key arguments. When adding a new argument, the annotator is asked to indicate the *stance* of the opinion (i.e., whether it is in support or against the related policy option). To facilitate this task, HyEnA highlights the most probable stance for the user as a label suggestion [35,36].

Topic Assignment We train a BERTopic model $\mathscr{T}$ on all the available opinions [37]. We create a short-list of topics, selected as the most frequent topics found by $\mathscr{T}$, with duplicates and unintelligible topics manually removed by two experts. We ask human annotators to associate the topics from the generated short-list to each argument. This topic assignment T is used in the second phase to compute argument similarity. Thus, in the first phase, HyEnA yields multiple key argument lists (one per annotator), each containing key arguments and their stances, and an assignment of key arguments to pre-selected topics.

3.3. Consolidation

In the first phase, (1) the annotators are exposed to a small subset of the opinions in the corpus, and (2) the interpretation of arguments is subjective. In the second phase, HyEnA seeks to *consolidate* the key argument lists generated in the first phase. Our goal is to increase the diversity of the resulting arguments and compensate for individual biases.

First, we create the union of all lists of key arguments generated in the first phase of HyEnA. Then, we ask the annotators to evaluate the similarity of the key argument pairs in the union list. Based on the similarity labels, we employ a clustering algorithm to group similar key arguments, producing a consolidated list of key arguments.

Pairwise Annotation To simplify the consolidation task, we present to the annotators one pair of key arguments at a time and ask whether the concepts described by the key arguments in the pair are semantically similar. To reduce human effort, we select only the most informative key argument pairs for manual annotation and automatically annotate the remaining pairs. To select the most informative pairs, we adapt the Partial-Ordering approach, POWER [38], as described below.

Let p_{ij} be a pair of key arguments $\langle a_i, a_j \rangle$. The similarity between the two key arguments in the pair is described by two *similarity scores*, s^1_{ij} and s^2_{ij}. By using multiple scores, we seek to make the similarity computation robust. For each p_{ij}, we compute the two similarity scores described in Table 2. We use cosine similarity for s^1_{ij} since the angular distance describes the semantic textual similarity between two arguments. In contrast, we use Euclidean distance for s^2_{ij} since the absolute values of the topic assignment are relevant.

Table 2. The similarity scores between key argument pairs used to create the pairwise dependency graph.

Measure	Description
$s^1_{ij} = \frac{\mathbf{i} \cdot \mathbf{j}}{\|\mathbf{i}\|\|\mathbf{j}\|}$	Cosine similarity between embeddings $\mathbf{i} = M_S(a_i)$ and $\mathbf{j} = M_S(a_j)$
$s^2_{ij} = \frac{1}{d(T(a_i), T(a_j))}$	Inverse of the Euclidean distance d between manual topic assignments T of a_i and a_j

Given the similarity scores, we construct a dependency graph G (as in the top-left part of Figure 2), where each key argument pair is a node in G and the edges indicate a Pareto dependency ($\succ$) between two pairs as follows:

$$p_{ij} \succeq p_{i'j'} \qquad \text{if} \quad \forall n \quad s^n_{ij} \geq s^n_{i'j'} \tag{1}$$

$$p_{ij} \succ p_{i'j'} \qquad \text{if} \quad p_{ij} \succeq p_{i'j'} \quad \text{and} \quad \exists n \quad s^n_{ij} > s^n_{i'j'} \tag{2}$$

Next, we follow POWER to extract disjoint paths from G. The highlighted path in the bottom-left part of Figure 2 is an example disjoint path. For every path, we perform a pairwise annotation as in the right part of Figure 2. We select the vertex at the middle of the unlabeled portion of the path and ask multiple (7) humans to indicate whether the concepts described by the two arguments in the pair are similar on a binary scale, and select the label with the majority vote. Given the annotation, we can automatically label (1) all following pairs in the path as similar (green) in case the vertex is labeled as similar or (2) all preceding pairs in the path as non-similar (red) in case the vertex is labeled as non-similar. In essence, using the Pareto dependency, we search for threshold similarity scores for each path, above which all pairs are considered similar, and below which all pairs are non-similar. Because this is a local threshold, we prevent over-generalization. To annotate the complete graph efficiently, we employ the parallel Multi-Path annotation algorithm [38].

Clustering Given a similarity label for each key argument pair, our goal is to identify groups of similar key arguments. However, the similarity among key arguments may not be transitive—given $\langle a_1, a_2 \rangle$ as similar and $\langle a_2, a_3 \rangle$ as similar, $\langle a_1, a_3 \rangle$ may be labeled as dissimilar. This can happen because (1) the interpretation of similarity can be subjective (for manually labeled pairs), and (2) the automatic approach is not always accurate (for

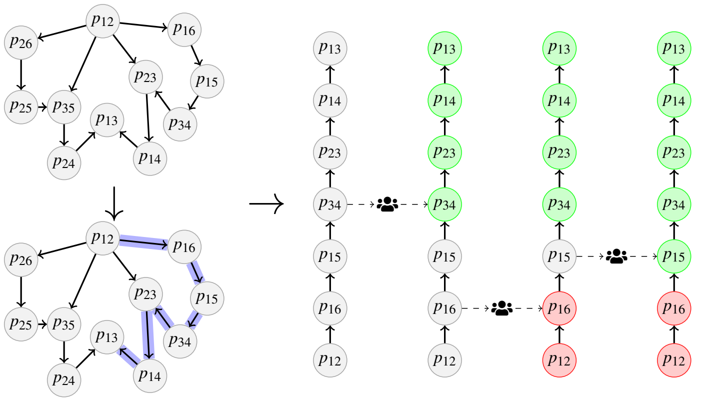

Figure 2. Pairwise annotation of the dependency graph, combining human and automatic judgements.

automatically labeled pairs). Thus, we employ a clustering algorithm for identifying a consolidated list. First, we construct a similarity graph, where each key argument is a node and there is an edge between two arguments if they are labeled as similar. Then, we employ out-of-the-box graph clustering algorithms for constructing argument clusters. These clusters form the *key argument lists*.

4. Experimental Setup

We involve 348 Prolific (`www.prolific.co`) crowd workers as annotators to evaluate HyEnA. We required the workers to be fluent in English, have an approval rate above 95%, and have completed at least 100 submissions. Our experiment was approved by an Ethics Committee and we received informed consent from each subject. We provide supplemental material, containing instructions provided to the annotators, our experiment protocol, experiment data, analysis code, and further details on the experiment (including inter-rater agreement scores) [39].

Table 3 shows an overview of the tasks in the experiment. First, we ask annotators to perform the HyEnA method to generate lists of key arguments for three corpora. Then, we compare the quality of the obtained lists of key arguments with lists generated for the same corpora via two baselines. All tasks except topic generation were performed by the crowd workers.

4.1. Phase 1: Key Argument Annotation

In the first phase of HyEnA, each annotator extracts a key arguments list from an opinion corpus. In each corpus, five annotators annotated 51 opinions each, for a total of 255 opinions. Of the 51 opinions, the first is selected randomly, and the following 50 are selected by FFT. This number of opinions was empirically selected to make the annotation feasible within a maximum of one hour.

Topics We train a BERTopic model on each opinion corpus, generating 59, 56, and 72 topics for the YOUNG, IMMUNE, and REOPEN corpora, respectively. Since the number

Table 3. Overview of the tasks in the experiment. Items to be annotated can be opinions (O), arguments (A), topics (T), or combinations.

Task	Option	# Items	# Annotators	Overlap
Key argument annotation	YOUNG	255 (O)	5	1
	IMMUNE	255 (O)	5	
	REOPEN	255 (O)	5	
Topic generation	all	45 (T)	2	2
Topic assignment	YOUNG	90 (A)	10	5
	IMMUNE	64 (A)	5	
	REOPEN	69 (A)	5	
Key argument consolidation	YOUNG	1538 (A+A)	99	3
	IMMUNE	824 (A+A)	57	
	REOPEN	940 (A+A)	87	
Key argument evaluation	YOUNG	172 (O+A)	28	7
	IMMUNE	133 (O+A)	21	
	REOPEN	157 (O+A)	21	

of resulting topics is too high for manual assignment of arguments to topics, we curate a short-list of topics per corpus. We select the 15 most frequent topics in a corpus and ask two experts, the first two authors, to remove duplicates (i.e., topics covering the same semantic aspect) and rate the clarity (i.e., how well the topic describes a relevant aspect of the discussion in the corpus) of each topic. Unique topics with an average clarity score above 2.5 compose the short-list of topics. Then, we ask crowd annotators to assign topics to each key argument generated in the first phase of HyEnA.

4.2. Phase 2: Key Argument Consolidation

In the second phase of HyEnA, we obtain similarity labels $y(a_i,a_j)$ (1 if similar, 0 if not) for all key argument pairs $\langle a_i,a_j\rangle$—some pairs are labeled by the annotators and others are automatically labeled. Given the similarity labels, we construct an argument similarity graph and cluster the graph to identify a consolidated list of key arguments.

Clustering We experiment with two well-known graph clustering algorithms: (1) Louvain clustering [40] uses network modularity to identify groups of vertices based on a resolution parameter r. (2) Self-tuning spectral clustering [41] uses dimensionality reduction in combination with k-means to obtain clusters, where k is the desired number of clusters. We select the parameters of these algorithms to minimize the error metric E shown in Eq. 3. The metric penalizes clusters having dissimilar argument pairs. That is, for a cluster $k \in K$ and $\forall a_i,a_j \in k$, if $y(a_i,a_j) = 1$, the error for that cluster is 0. If a cluster contains only a single element, we manually set the error for that cluster to 1, to discourage creating single-member clusters. For evaluation, our experts manually select a representative argument per key argument list, but other selection criteria based on argumentative quality or other in-cluster metrics can be used to streamline the process.

$$E = \frac{1}{|K|}\sum_{k\in K}\frac{\sum_{a_i,a_j\in k}\mathbb{1}_{y(a_i,a_j)=0}}{\binom{|k|}{2}} \tag{3}$$

4.3. Baselines and Evaluation Metrics

4.3.1. Comparison to Automated Baseline

We use the **ArgKP** argument matching model [13] to automatically extract key points from the corpus. ArgKP selects candidate key points from opinions using a manually-tuned heuristic, which filters opinions on their length, form, and predicted argument quality [34]. The original approach suggests relaxing its hyperparameters such that 20% of the opinions are selected as candidates. However, this caused substantially different candidates. Instead, we only relax the original hyperparameters slightly, causing $\sim$10% of opinions to be selected as key point arguments. Candidate key points and opinions are assigned a match score using a pretrained matching network based on RoBERTa [42]. Opinions only match the highest scoring candidate key points if their match score exceeds a threshold θ (corresponding to the BM+TH approach). After deduplication, this results in a single list of key arguments per option. We use two metrics, *coverage* (C) and *precision* (P), to compare HyEnA and ArgKP.

Coverage (C) is defined as the fraction of opinions mapped to an argument out of all the processed opinions [13]. To compute C, first, we extract the set of key arguments $\mathscr{A}_H$ from HyEnA based on opinions O_H^{obs} ($\subset O$) observed by the annotators. Further, if an argument is extracted from an observed opinion $o_i \in O_H^{obs}$, we add o_i to the set of *annotated* opinions O_H^{ann}. Similarly, we extract the set of key arguments $\mathscr{A}_A$ from ArgKP based on its observed set of opinions $O_A^{obs}(\equiv O)$, producing a set of *annotated* opinions O_A^{ann}. Then, the coverage with respect to *all* observed opinions is:

$$C_H = \frac{|O_H^{ann}|}{|O_H^{obs}|} \qquad (4) \qquad\qquad C_A = \frac{|O_A^{ann}|}{|O_A^{obs}|} \qquad (5)$$

Comparing coverages as defined above may not be fair since the set of observed opinions (i.e., the denominators of Eqs. 4 and 5) are not the same for HyEnA and ArgKP. Thus, we also compute coverage with respect to a set of *common* opinions, $O_H^{obs} \cap O_A^{obs}$, observed by both methods, as:

$$C_H^{common} = \frac{|O_H^{ann} \cap O_A^{obs}|}{|O_H^{obs} \cap O_A^{obs}|} \qquad (6) \qquad\qquad C_A^{common} = \frac{|O_A^{ann} \cap O_H^{obs}|}{|O_H^{obs} \cap O_A^{obs}|} \qquad (7)$$

We add the same term to both denominator and numerator in Eqs. 6 and 7 so that the coverage stays in the range [0, 1]. Note that $C_H^{common} = C_H$ since $O_H^{obs}, O_H^{ann} \subset O_A^{obs}(\equiv O)$.

Precision (P) is the fraction of mapped opinions for which the mapping is correct [13]. Thus, we must map a set of opinions to arguments in order to compute precision. For this mapping, we select the common opinions, $O_H^{ann} \cap O_A^{ann}$, that are annotated in both HyEnA and ArgKP. Then for each $o_i \in O_H^{ann} \cap O_A^{ann}$, we create two pairs $\langle o_i, \mathscr{A}_H(o_i)\rangle$ and $\langle o_i, \mathscr{A}_A(o_i)\rangle$, where $\mathscr{A}_H(o_i)$ and $\mathscr{A}_A(o_i)$ are the arguments associated with o_i by HyEnA and ArgKP, respectively. Then, we ask annotators to label $z(o_i, a_i) = 1$ for all matching pairs and $z(o_i, a_i) = 0$ for all non-matching pairs, and keep the majority consensus from multiple annotators. Given the opinion-argument mapping, we compute precision as:

$$P_H^{common} = \frac{\sum_{o_i \in O_H^{ann} \cap O_A^{ann}} z(o_i, \mathscr{A}_H(o_i))}{|O_H^{ann} \cap O_A^{ann}|} \quad (8) \qquad P_A^{common} = \frac{\sum_{o_i \in O_H^{ann} \cap O_A^{ann}} z(o_i, \mathscr{A}_A(o_i))}{|O_H^{ann} \cap O_A^{ann}|} \quad (9)$$

4.3.2. Comparison to Manual Baseline

A manual analysis involving six experts analyzed the feedback from a sample of participants (2,237 out of 26,293) over all policy options and identify key arguments [3]. However, they do not report the exact number of opinions analyzed. Since there are 36,781 opinions for the three options we analyze (Table 1), we estimate the number of opinions the six experts would have analyzed to be 3,129 across the three options. In contrast, HyEnA annotators analyze 765 intelligently selected opinions across the three options.

It is evident that HyEnA reduces the number of opinions analyzed. Further, we investigate the extent to which the key argument lists generated by HyEnA and the manual baseline have comparable insights. To do so, we report the number of HyEnA key arguments that are overlapping, missing, and new compared to the expert-identified key arguments. We cannot compute precision and coverage for the manual baseline because it does not include a mapping between key arguments and opinions.

5. Results and Discussion

Before comparing with the baselines, we analyze the intelligent sampling and merging techniques HyEnA employs in Phases 1 and 2.

5.1. Phase 1: Key Argument Annotation

Table 4 shows the number of different operations annotators perform in Phase 1. On average, the annotators identified 15 unique key arguments per option. About half of the opinions were skipped, mainly because the opinion lacked a clear argument. This is a positive result since the noise (i.e., irrelevant or non-argumentative opinions) in public feedback can be much higher. Thus, the argument quality classifier we incorporate for opinion sampling is effective in filtering noise. Further, the annotators marked only about 15% of the encountered opinions as already annotated key arguments, which shows that the FFT approach is effective in sampling a diverse set of opinions for annotation.

Table 4. The average annotation operations (and their standard deviation) in Phases 1 and 2.

	Phase 1			**Phase 2**	
Option	**# Args**	**# Skip**	**# Already**	Δ	τ
YOUNG	18.0 (5.5)	23.4 (5.4)	11.4 (9.0)	-61.6%	0.34
IMMUNE	12.8 (2.6)	31.4 (4.5)	8.6 (4.4)	-59.1%	0.42
REOPEN	13.8 (7.6)	29.2 (11.5)	10.2 (7.6)	-59.8%	0.41

5.2. Phase 2: Key Argument Consolidation

Table 4 also shows the benefit of POWER, HyEnA's approach for consolidating key arguments. The number of pairs requiring human annotation (Δ) was on average reduced by 60%. The transitivity score τ [43] indicates the extent to which transitivity holds among the similarity labels of argument pairs. The relatively low τ scores justify the subsequent clustering we perform. Louvain clustering yields the smallest error for the YOUNG and IMMUNE corpora, and spectral clustering for REOPEN corpus.

5.3. Comparison with Automated Baseline

Figure 3 compares the coverage and precision of HyEnA and ArgKP. The low coverage (for both methods) indicates that a large number of opinions do not map to a key argument. This is not surprising since real-world opinions are noisy.

Considering *all* observed opinions (C_H and C_A), HyEnA yields slightly higher coverage than ArgKP in the YOUNG and REOPEN corpora. In contrast, ArgKP yields higher coverage than HyEnA in the IMMUNE corpus. We attribute this to the repeated arguments in the IMMUNE corpus. As 83% of opinions are con-opinions, the IMMUNE policy option (Table 1) was highly opposed and its corpus contains many repeated arguments against that option. Since the set of *all* observed opinions is the entire corpus for ArgKP, the repeated arguments inflate its coverage. However, since HyEnA observes only a small subset of diverse opinions from the corpus, the repeated arguments do not influence its coverage significantly. Thus, we compare the coverage of HyEnA and ArgKP with respect to a *common* set of diverse opinions. In this comparison (C_H^{common} and C_A^{common}), HyEnA yields consistently higher coverage (0.34 on average) than ArgKP (0.16 on average) in all three corpora.

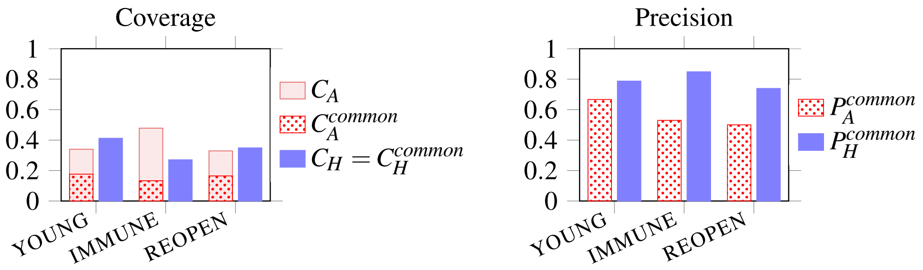

Figure 3. Comparing HyEnA and ArgKP.

ArgKP yields a larger number of key arguments (around 30 for each option) than HyEnA. However, these arguments lead to an average precision of 0.56. In contrast, HyEnA extracts fewer argument clusters (on average 17 per option), but with higher precision (0.80). Further, we notice that HyEnA annotators actively rephrase the content of the key arguments—only in 22% of the annotated key arguments, more than half of the key argument text is directly copied from the original opinion text; in contrast, the key points generated by ArgKP are composed of the original text.

5.4. Comparison with Manual Baseline

Table 5 shows counts of overlapping (yes, yes), missing (no, yes), and new (yes, no) key arguments between HyEnA and the manual baseline. HyEnA required an analysis of 765 opinions, compared to the estimated 3,000 opinions seen in the manual baseline. Despite the lower human effort, the HyEnA lists largely overlap with the expert lists.

Table 5. Confusion matrix, comparing the key argument lists of HyEnA and manual baseline.

			Manual baseline							
			yes	no		yes	no		yes	no
HyEnA	yes	YOUNG	8	7	IMMUNE	7	2	REOPEN	10	1
	no		1	–		0	–		4	–

HyEnA missed some key arguments that the experts identified, e.g., a key argument about building herd immunity was not in the HyEnA list for the REOPEN option. We conjecture that increasing the number of opinions annotated in HyEnA would subsequently yield the missing insights. HyEnA also led to new insights that experts missed, e.g., an argument about the physical well-being of young people was not on the expert list for the YOUNG option. Likely, the random sample of opinions experts analyzed did not include opinions supporting this argument, whereas the smaller set sampled in HyEnA did.

6. Conclusion and Directions

We develop and evaluate HyEnA, a hybrid method that combines human judgements with automated methods to generate a diverse set of key arguments. HyEnA extracts key arguments from noisy opinions and achieves consistent coverage, whereas the coverage of a state-of-the-art automated method drops by 50% when switching from all (containing repeated) opinions to diverse opinions. Moreover, the key arguments extracted by HyEnA are more precise than those extracted by the automated baseline. Additionally, HyEnA provides important insights that were not included in an expert-driven analysis of the same corpus, despite requiring fewer opinions to be analyzed.

The pairwise comparison in the consolidation phase is the most human-intensive task in HyEnA, and the effort increases with the number of analyzed opinions. Also, comparing arguments is cognitively demanding. HyEnA reduced the number of comparisons required in the consolidation phase by 60%. Additional research is necessary to reduce the consolidation effort further. For example, first clustering the key arguments and then consolidating the arguments within these clusters (reverse order as HyEnA) can influence the performance and effort but requires further investigation.

Finding arguments in a discourse is only one aspect that constitutes the perspectives in a discussion. Future work can incorporate analysis of other perspective factors, such as values [44,45], sentiment, emotion, and attribution [46]. By combining these rich aspects with arguments, we can merge the logical basis of the discussion with other semantic and syntactic information, allowing close scrutiny of the perspectives in opinions.

Acknowledgements This research was (partially) funded by the Hybrid Intelligence Center, a 10-year programme funded by the Dutch Ministry of Education, Culture, and Science through the Netherlands Organisation for Scientific Research.

References

[1] Kythreotis AP, Mantyka-Pringle C, Mercer TG, Whitmarsh LE, Corner A, Paavola J, et al. Citizen Social Science for More Integrative and Effective Climate Action: A Science-Policy Perspective. Frontiers in Environmental Science. 2019;7. Available from: https://www.frontiersin.org/article/10.3389/fenvs.2019.00010.

[2] Lee S, Hwang C, Moon MJ. Policy learning and crisis policy-making: quadruple-loop learning and COVID-19 responses in South Korea. Policy and Society. 2020;39(3):363-81. PMID: 35039726. Available from: https://doi.org/10.1080/14494035.2020.1785195.

[3] Mouter N, Hernandez JI, Itten AV. Public participation in crisis policymaking. How 30,000 Dutch citizens advised their government on relaxing COVID-19 lockdown measures. PLOS ONE. 2021 05;16(5):1-42. Available from: https://doi.org/10.1371/journal.pone.0250614.

[4] Lawrence J, Reed C. Argument Mining: A Survey. Computational Linguistics. 2020 01;45(4):765-818. Available from: https://doi.org/10.1162/coli_a_00364.

[5] Stab C, Miller T, Schiller B, Rai P, Gurevych I. Cross-topic Argument Mining from Heterogeneous Sources. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium: Association for Computational Linguistics; 2018. p. 3664-74. Available from: https://aclanthology.org/D18-1402.

[6] Thorn Jakobsen TS, Barrett M, Søgaard A. Spurious Correlations in Cross-Topic Argument Mining. In: Proceedings of *SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics. Online: Association for Computational Linguistics; 2021. p. 263-77. Available from: https://aclanthology.org/2021.starsem-1.25.

[7] Reimers N, Schiller B, Beck T, Daxenberger J, Stab C, Gurevych I. Classification and Clustering of Arguments with Contextualized Word Embeddings. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: Association for Computational Linguistics; 2019. p. 567-78. Available from: https://aclanthology.org/P19-1054.

[8] Ein-Dor L, Shnarch E, Dankin L, Halfon A, Sznajder B, Gera A, et al. Corpus Wide Argument Mining—A Working Solution. Proceedings of the AAAI Conference on Artificial Intelligence. 2020 Apr;34(05):7683-91. Available from: https://ojs.aaai.org/index.php/AAAI/article/view/6270.

[9] Klein M. Enabling Large-Scale Deliberation Using Attention-Mediation Metrics. Computer Supported Cooperative Work (CSCW). 2012;21(4-5):449-73.

[10] Price V. Social Identification and Public Opinion: Effects of Communicating Group Conflict. Public Opinion Quarterly. 1989 01;53(2):197-224. Available from: https://doi.org/10.1086/269503.

[11] Schulz-Hardt S, Frey D, Lüthgens C, Moscovici S. Biased information search in group decision making. Journal of Personality and Social Psychology. 2000;78(4):655-69. Available from: https://doi.org/10.1037%2F0022-3514.78.4.655.

[12] Bar-Haim R, Eden L, Friedman R, Kantor Y, Lahav D, Slonim N. From Arguments to Key Points: Towards Automatic Argument Summarization. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Online: Association for Computational Linguistics; 2020. p. 4029-39. Available from: https://aclanthology.org/2020.acl-main.371.

[13] Bar-Haim R, Kantor Y, Eden L, Friedman R, Lahav D, Slonim N. Quantitative argument summarization and beyond: Cross-domain key point analysis. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Online: Association for Computational Linguistics; 2020. p. 39-49. Available from: https://aclanthology.org/2020.emnlp-main.3.

[14] Cabrio E, Villata S. Five Years of Argument Mining: A Data-Driven Analysis. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence. IJCAI'18. AAAI Press; 2018. p. 5427–5433.

[15] Teruel M, Cardellino C, Cardellino F, Alonso Alemany L, Villata S. Increasing Argument Annotation Reproducibility by Using Inter-annotator Agreement to Improve Guidelines. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). Miyazaki, Japan: European Language Resources Association (ELRA); 2018. p. 4061-4. Available from: https://aclanthology.org/L18-1640.

[16] Niculae V, Park J, Cardie C. Argument Mining with Structured SVMs and RNNs. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Pa-

pers). Vancouver, Canada: Association for Computational Linguistics; 2017. p. 985-95. Available from: https://aclanthology.org/P17-1091.

[17] Eger S, Daxenberger J, Gurevych I. Neural End-to-End Learning for Computational Argumentation Mining. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vancouver, Canada: Association for Computational Linguistics; 2017. p. 11-22. Available from: https://aclanthology.org/P17-1002.

[18] Freeman JB. Argument structure:. 2011th ed. Argumentation Library. Dordrecht, Netherlands: Springer; 2011.

[19] Daxenberger J, Eger S, Habernal I, Stab C, Gurevych I. What is the Essence of a Claim? Cross-Domain Claim Identification. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. Copenhagen, Denmark: Association for Computational Linguistics; 2017. p. 2055-66. Available from: https://aclanthology.org/D17-1218.

[20] Stab C, Gurevych I. Annotating Argument Components and Relations in Persuasive Essays. In: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers. Dublin, Ireland: Dublin City University and Association for Computational Linguistics; 2014. p. 1501-10. Available from: https://aclanthology.org/C14-1142.

[21] Biran O, Rambow O. Identifying Justifications in Written Dialogs. In: 2011 IEEE Fifth International Conference on Semantic Computing; 2011. p. 162-8.

[22] Conneau A, Kiela D, Schwenk H, Barrault L, Bordes A. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. Copenhagen, Denmark: Association for Computational Linguistics; 2017. p. 670-80. Available from: https://aclanthology.org/D17-1070.

[23] Reimers N, Gurevych I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). Hong Kong, China: Association for Computational Linguistics; 2019. p. 3982-92. Available from: https://aclanthology.org/D19-1410.

[24] Xu J, He X, Li H. Deep Learning for Matching in Search and Recommendation. Foundations and Trends® in Information Retrieval. 2020;14(2–3):102-288. Available from: http://dx.doi.org/10.1561/1500000076.

[25] Zhong M, Liu P, Chen Y, Wang D, Qiu X, Huang X. Extractive Summarization as Text Matching. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Online: Association for Computational Linguistics; 2020. p. 6197-208. Available from: https://aclanthology.org/2020.acl-main.552.

[26] Howard J, Ruder S. Universal Language Model Fine-tuning for Text Classification. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Melbourne, Australia: Association for Computational Linguistics; 2018. p. 328-39. Available from: https://aclanthology.org/P18-1031.

[27] McCoy T, Pavlick E, Linzen T. Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: Association for Computational Linguistics; 2019. p. 3428-48. Available from: https://aclanthology.org/P19-1334.

[28] Akata Z, Balliet D, de Rijke M, Dignum F, Dignum V, Eiben G, et al. A Research Agenda for Hybrid Intelligence: Augmenting Human Intellect With Collaborative, Adaptive, Responsible, and Explainable Artificial Intelligence. Computer. 2020;53(8):18-28.

[29] Habernal I, Gurevych I. Argumentation Mining in User-Generated Web Discourse. Computational Linguistics. 2017 Apr;43(1):125-79. Available from: https://aclanthology.org/J17-1004.

[30] Sennrich R, Haddow B, Birch A. Improving Neural Machine Translation Models with Monolingual Data. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Berlin, Germany: Association for Computational Linguistics; 2016. p. 86-96. Available from: https://aclanthology.org/P16-1009.

[31] Eger S, Daxenberger J, Stab C, Gurevych I. Cross-lingual Argumentation Mining: Machine Translation (and a bit of Projection) is All You Need! In: Proceedings of the 27th International Conference on Computational Linguistics. Santa Fe, New Mexico, USA: Association for Computational Linguistics; 2018. p. 831-44. Available from: https://aclanthology.org/C18-1071.

[32] Daza A, Frank A. X-SRL: A Parallel Cross-Lingual Semantic Role Labeling Dataset. In: Proceedings of

the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Online: Association for Computational Linguistics; 2020. p. 3904-14. Available from: https://aclanthology.org/2020.emnlp-main.321.

[33] Basu S, Banerjee A, Mooney RJ. 32. In: Active Semi-Supervision for Pairwise Constrained Clustering. Society for Industrial and Applied Mathematics; 2004. p. 333-44. Available from: https://epubs.siam.org/doi/abs/10.1137/1.9781611972740.31.

[34] Gretz S, Friedman R, Cohen-Karlik E, Toledo A, Lahav D, Aharonov R, et al. A Large-Scale Dataset for Argument Quality Ranking: Construction and Analysis. Proceedings of the AAAI Conference on Artificial Intelligence. 2020 Apr;34(05):7805-13. Available from: https://ojs.aaai.org/index.php/AAAI/article/view/6285.

[35] Schulz C, Meyer CM, Kiesewetter J, Sailer M, Bauer E, Fischer MR, et al. Analysis of Automatic Annotation Suggestions for Hard Discourse-Level Tasks in Expert Domains. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: Association for Computational Linguistics; 2019. p. 2761-72. Available from: https://aclanthology.org/P19-1265.

[36] Beck T, Lee JU, Viehmann C, Maurer M, Quiring O, Gurevych I. Investigating label suggestions for opinion mining in German Covid-19 social media. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). Online: Association for Computational Linguistics; 2021. p. 1-13. Available from: https://aclanthology.org/2021.acl-long.1.

[37] Grootendorst M. BERTopic: Neural topic modeling with a class-based TF-IDF procedure. arXiv; 2022. Available from: https://arxiv.org/abs/2203.05794.

[38] Chai C, Li G, Li J, Deng D, Feng J. Cost-Effective Crowdsourced Entity Resolution: A Partial-Order Approach. In: Proceedings of the 2016 International Conference on Management of Data. SIGMOD '16. New York, NY, USA: Association for Computing Machinery; 2016. p. 969–984. Available from: https://doi.org/10.1145/2882903.2915252.

[39] van der Meer M, Liscio E, Jonker CM, Plaat A, Vossen P, Murukannaiah PK. HyEnA: A Hybrid Method for Extracting Arguments from Opinions: Supplementary Material; 2022. https://liacs.leidenuniv.nl/~meermtvander/publications/hyena/.

[40] Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E. Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment. 2008 oct;2008(10):P10008. Available from: https://doi.org/10.1088/1742-5468/2008/10/p10008.

[41] Zelnik-Manor L, Perona P. Self-Tuning Spectral Clustering. In: Proceedings of the 17th International Conference on Neural Information Processing Systems. NIPS'04. Cambridge, MA, USA: MIT Press; 2004. p. 1601–1608. Available from: https://proceedings.neurips.cc/paper/2004/file/40173ea48d9567f1f393b20c855bb40b-Paper.pdf.

[42] Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, et al. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:190711692. 2019.

[43] Newman MEJ, Watts DJ, Strogatz SH. Random graph models of social networks. Proceedings of the National Academy of Sciences. 2002;99(suppl_1):2566-72. Available from: https://www.pnas.org/doi/abs/10.1073/pnas.012582999.

[44] Liscio E, van der Meer M, Siebert LC, Jonker CM, Mouter N, Murukannaiah PK. Axies: Identifying and Evaluating Context-Specific Values. In: Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems. AAMAS '21. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems; 2021. p. 799–808.

[45] Liscio E, van der Meer M, Siebert LC, Jonker CM, Murukannaiah PK. What Values should an Agent Align with? An Empirical Comparison of General and Context-Specific Values. Autonomous Agents and Multi-Agent Systems. 2022;36(1):23.

[46] van Son C, Caselli T, Fokkens A, Maks I, Morante R, Aroyo L, et al. GRaSP: A Multilayered Annotation Scheme for Perspectives. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16). Portorož, Slovenia: European Language Resources Association (ELRA); 2016. p. 1177-84. Available from: https://aclanthology.org/L16-1187.