Universiteit
Leiden
The Netherlands

Opinion diversity through hybrid intelligence

Meer, M.T. van der

Citation
Meer, M. T. van der. (2025, March 26). *Opinion diversity through hybrid intelligence*. *SIKS Dissertation Series*. Retrieved from https://hdl.handle.net/1887/4209024

Version:	Publisher's Version
License:	Licence agreement concerning inclusion of doctoral thesis in the Institutional Repository of the University of Leiden
Downloaded from:	https://hdl.handle.net/1887/4209024

Note: To cite this publication please use the final published version (if applicable).

4

A Hybrid Intelligence Method for Argument Mining

Large-scale survey tools enable the collection of citizen feedback in opinion corpora. Extracting the key arguments from a large and noisy set of opinions helps in understanding the opinions quickly and accurately. Fully automated methods can extract arguments but (1) require large labeled datasets that induce large annotation costs 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 citizen 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 to 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 artificial intelligence.

Michiel van der Meer, Enrico Liscio, Aske Plaat, Piek Vossen, Catholijn M. Jonker, and Pradeep K. Murukannaiah. 2024. A Hybrid Intelligence Method for Argument Mining. In *Journal of Artificial Intelligence Research* 80, pages 1187–1222.

4.1 Introduction

To make decisions on large public issues, such as combating a pandemic and transitioning to green energy, policymakers often turn to the citizens for feedback [215, 226]. This feedback provides insights into public opinion and contains viewpoints from many individuals with different perspectives. Involving the public in the decision-making process helps in gaining their support when the decisions are to be implemented, fostering the legitimacy of the process [292].

In the face of crises, decisions must be made swiftly. Thus, collecting feedback, analyzing it, and making recommendations ought to be performed under tight time constraints. For example, when deciding on relaxing COVID-19 measures in the Netherlands, researchers had one month to design the experiment, collect public feedback, and make recommendations to the government [274]. The time constraint limits the amount of information researchers can analyze, potentially painting an incomplete picture of the opinions. In the scenario above, researchers processed data manually and they could only analyze less than 8% of the qualitative feedback provided by more than 25,000 participants.

Argument Mining (AM) [224] methods can assist in increasing the efficiency of feedback analysis by, e.g., locating and interpreting argumentative feedback and classifying statements as supporting or opposing a decision. However, applying automated AM methods for feedback analysis poses three main challenges. First, AM methods generalize poorly across domains [367, 382, 405]. Thus, they require large amounts of domain-specific training data, which is often not available. The use of pretrained language models, with the pre- or fine-tuning paradigm, mitigates but does not solve the reliance on large domain-specific training datasets [112, 315]. Second, although AM methods can identify argumentative content, they often do not compress the information [68, 93, e.g.]. That is, they struggle to recognize whether two arguments describe the same point of view, leaving the policymakers with the significant manual labor of aggregating arguments [209, 210]. Finally, naively relying on a small sample of labeled data might cause minority opinions to be ignored as they are not well represented [204], creating a bias toward popular (repeated) arguments, which can perpetuate echo chambers and filter bubbles [307, 342].

The *key point analysis* (KPA) task [32] 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 [33]. However, the reliance on a few human expert annotators introduces biases of the human experts and may not be representative of the opinions of the larger population. This defeats the purpose of engaging the larger public in a bottom-up deliberative decision-making process.

We argue for a crowd-sourced human-machine approach for argument extraction, combining the scalability of automated methods and the human understanding of others' perspectives. 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*, *consolidation*, and *selection* phases. HyEnA employs human (crowd) annotators and supports them via intelligent algorithms based on natural language processing (NLP) techniques for analyzing opinions provided by a large audience, as shown in Figure 4.1.

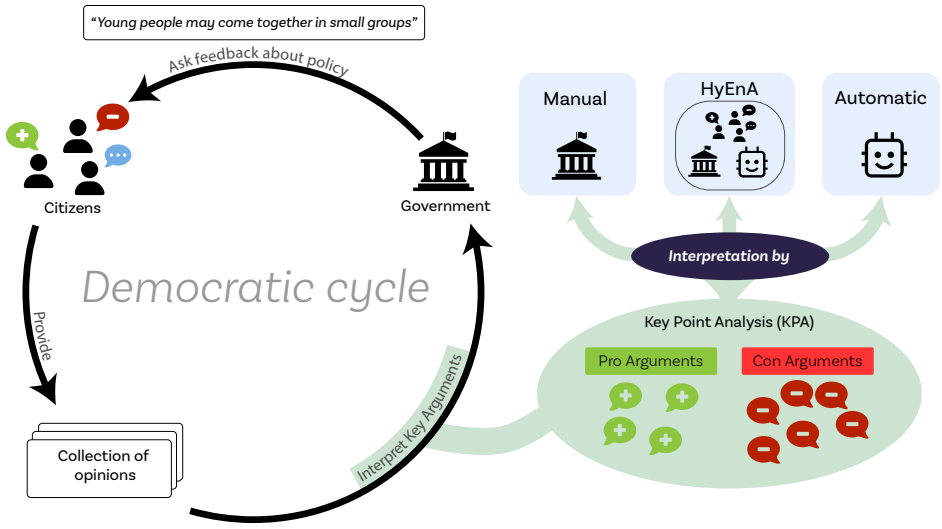

Figure 4.1: In a democratic cycle, citizens provide their opinions on options for governmental decision-making and their opinions need to be interpreted. Insights into the arguments embedded in their comments can be provided by Key Point Analysis (KPA). To perform KPA, most analysis is performed either manually or automatically. In our work, we propose HyEnA, a hybrid method.

HyEnA is evaluated on three corpora, each containing more than 10K public opinions on relaxing COVID-19 restrictions [274]. We compare HyEnA with an automated approach [33] performing the KPA task. In addition, we compare the key arguments generated by HyEnA with manually obtained insights identified by experts [274]. We find that HyEnA outperforms the automated baseline in terms of precision and diversity, specifically when confronted with a set of varied perspectives. HyEnA also yields better results than manual analysis, as fewer opinions needed to be analyzed in order to obtain a wider set of key arguments.

Contributions (1) We present a hybrid method for key argument extraction, which generates a diverse set of key arguments from a collection of opinionated user comments. (2) We evaluate our method on real-world 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. (3) We extensively discuss the implications of incorporating recent advances in NLP, such as Large Language Models (LLMs), into the workflow of our hybrid method.

Extension In this Chapter, we provide details on an extended version of the HyEnA method [398, 403]. The original HyEnA method outputs argument clusters, and leverages manual annotations from the first two phases to select arguments from argument clusters. The extension introduces a method for selecting the most representative argument from each clus-

ter through *argument selection*. The need to summarize argument clusters is not specific to HyEnA, as previous AM applications also retrieve clusters instead of singular arguments [54, 93, 417]. We compare various techniques to accomplish this task, including generative large language models. Furthermore, we run additional experiments to demonstrate how the new argument selection step can be incorporated into the HyEnA pipeline, and rerun the original evaluation to compare between HyEnA with and without the inclusion of argument selection. Finally, we perform additional analyses to derive further insights from annotators in HyEnA. We also provide our code, annotation guidelines, and experimental details in the supplementary materials [404].

Structure Section 4.2 provides background on Argument Mining for public opinions, and Section 4.3 introduces the HyEnA method for extracting arguments. We outline the experimental setup in Section 4.4 and provide extensive results in Section 4.5. A discussion of our work is given in Section 4.6 and we conclude with Section 4.7.

4.2 Related work

We describe related work on Argument Mining, methods for summarizing arguments, and their application to opinion analysis.

4.2.1 Computational Argument Analysis

Argument Mining (AM) methods [62, 224] focus on the recognition, extraction, and computational analysis of arguments presented in natural language. They seek to discover arguments brought forward by speakers and identify connections between them. Typically, AM techniques concern themselves with finding the *structure* of arguments [407], with the goal of finding premises for supporting or refuting conclusions.

AM is a challenging problem. The ability to recognize and extract arguments from text (for humans and machines, alike) is dependent on the argumentativeness of the underlying data. Often, significant effort is required by human annotators to reach moderate inter-rater agreement when annotating arguments [381]. Given argumentative texts, modern NLP models are reasonably good at recognizing argumentative discourse within specific contexts [110, 285, 315].

Typically, the first step of AM is to identify the elemental components of arguments (e.g., *claims* and *premises*) in text [296]. The combination of such components forms a structured argument. However, there is currently no consensus on the exact linguistic notion of such elemental components, with multiple levels of granularity being proposed [47, 92, 129, 418]. Nonetheless, a few characteristics have been recognized as important for recognizing arguments, namely that arguments (1) contain (informal) logical reasoning [365], (2) address a *why* question [50], and (3) have a non-neutral stance towards the issue being discussed [365].

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 through automation.

4.2.2 Summarization of Arguments

Automated methods have been proposed to derive high-level insights from large-scale argumentative content. For instance, these approaches focus on indexing and searching through arguments [366, 439], or creating visual overviews of argument structures [63, 197]. While these may provide access to argumentative content, they are limited in providing a single high-level overview of the arguments on a topic of discussion. Instead, we turn our focus to approaches that create a comprehensible text-based summary from a large corpus of individual comments [33, e.g.]. 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 [32]. Such approaches have been applied in multiple domains, showcasing their applicability across context [34] at varying levels of granularity [66]. While these approaches present high-level arguments, they struggle to capture diversity in opinions, which is important for accommodating multiple perspectives [405]. In this work, we evaluate the performance of these approaches on a novel domain of COVID-19 measures and compare it against HyEnA.

Additionally, there exists an extended 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 [81, 314]. They have made a significant impact across a range of tasks [442, 453]. However, downstream applications often need additional fine-tuning [172] in order to perform a task well. They also capture generic aspects of semantic similarity and entailment, which may not be applicable to arguments [314], or overfit to spurious patterns in the data [262]. Thus, such methods require significant adaptation to effectively compress information in particular domains. Recently, Large Language Models (LLMs) have been shown to perform well on inference tasks with out-of-distribution data [419]. However, we argue that a plurality of (human) perspectives is necessary to perform sensitive tasks such as the summarization of arguments, which may in turn be used to inform policy-makers about the sentiment of a population [378]. Yet, LLMs might be adequate for specific subtasks, as we showcase in the third phase of the HyEnA method.

4.3 Method

HyEnA is a hybrid method since it combines automated techniques and human judgment [5, 97]. HyEnA guides human annotators in synthesizing *key arguments* (i.e., high-level semantically distinct arguments that describe relevant aspects of the topic under discussion) from an *opinion corpus* composed of individual *opinions* (textual comments) on a topic. Key arguments are high-level and summarize a group of arguments, similar to key points as introduced by [32]. We adopt the term key argument, to emphasize their argumentative nature, as opposed to more generic extractive summarization [346, e.g.].

HyEnA consists of three phases (Figure 4.2). In the first phase (*Key Argument Annotation*), an intelligent sampling algorithm guides human annotators individually through an opinion corpus to extract high-level information from the opinions. In the second and third phases, HyEnA aims to reduce the subjectivity in the first phase annotations by combining and rewriting arguments that were individually annotated. In the second phase (*Key Argument Consolidation*), an intelligent merging strategy supports a new group of annotators in merging the results from the first phase into clusters of arguments, combining manual and

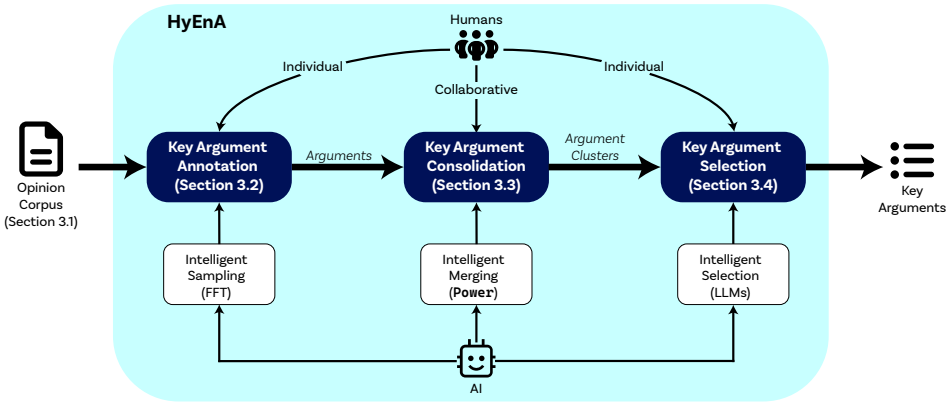

Figure 4.2: Overview of the HyEnA method.

automatic labeling. In the third phase (*Key Argument Selection*), HyEnA employs an automated method to synthesize a single argument that represents the arguments belonging to the same merged argument cluster. The final output of HyEnA is a list of key arguments grounded on the opinions in the corpus.

4.3.1 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 [274]. In a PVE, participants are offered a set of policy options and asked to select their preferred portfolio of choices. Then, the participants are asked to explain 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. The data used in our experiments concerns the COVID-19 regulations in the Netherlands during the height of the pandemic, in May 2020. We chose this scenario because (1) we had access to a unique dataset of citizen-provided comments on COVID-19 regulations, (2) we were able to run the study while the topic was still relevant, making it interesting for crowd workers, (3) a manual analysis had been performed over the exact same data, allowing for comparison to a human-only baseline, and (4) the data is reflective of real-world conditions, e.g. feedback was obtained in a matter of days but contains input from a broad group of citizens encompassing broad demographics. We analyze feedback from 26,293 Dutch citizens on three policy options, treating comments on each option as an opinion corpus. Table 4.1 shows examples of opinions provided for each different policy option. In our experiments, the HyEnA method is applied to one corpus at a time. Since we use data from a publicly run citizen feedback experiment, we observe that some options attracted more pro comments than others. We picked these three options with different pro/con ratios to investigate their impact on the key argument extraction task. The opinions in these corpora are similar to noisy user-generated web comments [156], may span multiple sentences, and contain more than one argument at a time. For each policy option, we use the keyword in uppercase as the option identifier in the remainder of the chapter.

The original opinions were provided in Dutch. To accommodate a diverse set of anno-

Policy option	Example opinion	Num. Opinions	Pro/Con Ratio
YOUNG people may come together in small groups	Then they can go back to school (Pro)	13400	0.66/0.34
All restrictions are lifted for persons who are IMMUNE	Encourages inequality (Con)	10567	0.17/0.83
REOPEN hospitality and entertainment industry	The economic damage is too high (Pro)	12814	0.55/0.45

Table 4.1: Example opinions in the COVID-19 corpora. The collection of opinions for a policy option forms an opinion corpus.

tators 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 [94, 111, 349]. Although all experiments are conducted in English, the link to the original Dutch text is preserved for future applications.

4.3.2 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) [314]. 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 [37]. 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 [144] to select one single clearest opinion related to the policy option. In this way, we aim to increase 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 [42, 341].

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

Table 4.2: The similarity scores between key argument pairs used to create the pairwise dependency graph.

Topic Assignment We use a BERTopic [147] model $\mathcal{T}$ to extract clusters of topics from the corpus. We train $\mathcal{T}$ on all opinions in the corpus and select the most frequent topics found by $\mathcal{T}$, with duplicates and unintelligible topics manually removed by two experts. We ask a new set of human annotators, different from those in argument extraction, to associate the topics from the generated shortlist with each argument, resulting in an n-hot vector for each argument a per annotator. We obtain the final topic assignment T by summing over all annotators. 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 pre-selected topics to key arguments.

4.3.3 Key Argument 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, the annotators are presented with one pair of key arguments at a time and asked whether the concepts described by the key arguments in the pair are 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 adopt a Partial-Ordering approach, POWER [67], 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 4.2. 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.

Given the similarity scores, we construct a dependency graph G (as in the top-left part of Figure 4.3), where each key argument pair is a vertex in G and the edges indicate a Pareto dependency ($\succ$) between two pairs—the direction of the edge points to the argument pair with greater similarity. A Pareto dependency holds if one of the two scores is strictly greater,

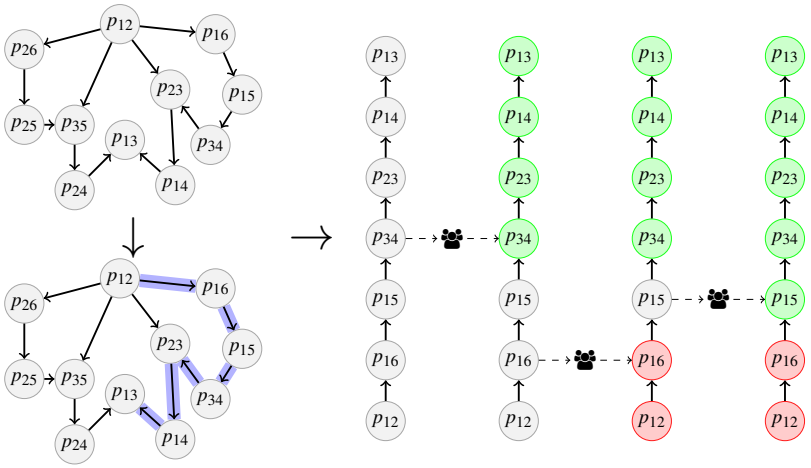

Figure 4.3: Pairwise annotation of the dependency graph, combining human and automatic judgments. Vertices indicate argument pairs; the edge direction points to the argument pair with greater similarity. The highlighted blue edges are a disjoint path selected by the Power algorithm. Iteratively, vertices are annotated as similar (green) or non-similar (red).

with all others being at least equal between two arguments. We define the dependency as follows:

$$p_{ij} \succeq p_{i'j'} \quad \text{if} \quad \forall n \quad s^n_{ij} \geq s^n_{i'j'} \tag{4.1}$$

$$p_{ij} \succ p_{i'j'} \quad \text{if} \quad p_{ij} \succeq p_{i'j'} \quad \text{and} \quad \exists n \quad s^n_{ij} > s^n_{i'j'} \tag{4.2}$$

Next, we follow Power to extract disjoint paths from G. The highlighted path in the bottom-left part of Figure 4.3 is an example disjoint path. For every path, we perform a pairwise annotation as in the right part of Figure 4.3. We select the vertex at the middle of the unlabeled portion of the path and ask up to seven humans to indicate whether the concepts described by the two arguments in the pair are similar on a binary scale. The arguments are similar when they are essentially bringing up the same point, i.e. provide the same reasoning. We select the label with the majority vote. Given the annotation, we can automatically label (1) all following pairs in the path as similar (yellow) 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 [67].

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 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 vertex 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*.

4.3.4 Key Argument Selection

In the third step of HyEnA, we extract a single argument from each cluster, obtaining the final list of key arguments for the opinion corpus. Formally, for every cluster $k \in K$, we create an argument a_k that is *representative* of that cluster. Argument selection methods can be extractive (select an argument from the cluster) or abstractive (generate a new argument that summarizes the cluster). Since there are many methods available for selecting arguments, we can experiment with multiple, and pick the best-performing method. In that case, we again pick an intermediate evaluation metric, which we use to select the best selection method. While there is no human annotation involved in this step, we still consider this higher-level algorithmic design a hybrid process, and thus a collaboration between humans and AI. For the task of selecting relevant arguments, we compare the following four types of approaches.

Centroids For every cluster k, we compute a sentence embedding of every argument a_k using M_S. Then, we compute pairwise distances between all arguments inside the same cluster. We select the argument with the lowest average distance, measured using cosine similarity, to all other arguments.

Argument Quality We use a model that measures argument quality to select the argument from each cluster with the highest quality. We use the same argument classifier as in the Key Argument Annotation phase, trained on the ArgQ dataset [144].

Prompting We prompt an LLM to synthesize a single argument out of the arguments provided in the argument cluster [58]. We experiment with an open-source and a closed-source model.

Random As a baseline, we randomly select an argument from the cluster to represent the entire argument cluster.

4.4 Experimental Setup

We involve 378 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, experiment protocol, experiment data, analysis code, and additional details on the experiment [404].

Table 4.3 shows an overview of the tasks in the experiment. First, we ask annotators to perform the HyEnA method to generate key argument lists for three corpora. Then, we compare the quality of the obtained lists with lists generated for the same corpora via two baselines. All tasks except topic generation were performed by the crowd workers, with most

Task	Option	Num. Items	Num. Annotators	Num. Annotators per item
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	91 (A)	10	5
	IMMUNE	66 (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	248 (O+A)	42	7
	IMMUNE	193 (O+A)	29	
	REOPEN	221 (O+A)	29	

Table 4.3: Overview of the tasks in the experiment. Items to be annotated can be opinions (O), arguments (A), topics (T), or combinations. † denotes expert annotators.

of the task instances annotated by multiple annotators to investigate the agreement between annotators.

4.4.1 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 per corpus. 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. We instantiate the S-BERT model M_S using the Huggingface Model Hub[1]. Since our opinion corpus stems from the PVE procedure, we have explicit labels denoting whether a comment was left in favor (*pro*) or opposing (*con*) a proposed policy, which we leverage for the argument stance label suggestion. For obtaining argument quality scores, we use the IBM API [35] to avoid having to retrain a new model.

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 of resulting topics is too high for the 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

[1] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

Method	Model	Type	Open	Size
Random	–	extractive	–	–
Centroid	S-BERT	extractive	yes	22M
Prompting	ChatGPT	abstractive	no	175B
	Llama	abstractive	yes	7B
Quality	ArgQ	extractive	no	125M

Table 4.4: Argument selection algorithms.

the corpus) of each topic. Unique topics with an average clarity score above 2.5 compose the shortlist of topics. Then, we ask crowd annotators to assign topics to each key argument generated in the first phase of HyEnA.

4.4.2 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 [52] uses network modularity to identify groups of vertices based on a resolution parameter r. (2) Self-tuning spectral clustering [446] 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. 4.3.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. We base E on the homogeneity metric [323], although we do not have access to the ground truth cluster assignments for each argument. Instead, we assume that if all manually labeled arguments are considered similar, they would have been assigned to a single cluster, resulting in a homogenous cluster.

$$E = \frac{1}{|K|} \sum_{k \in K} \frac{\sum_{a_i, a_j \in k} \mathbb{1}_{y(a_i, a_j)=0}}{\binom{|k|}{2}} \tag{4.3}$$

4.4.3 Phase 3: Key Argument Selection

In the third phase, we use a mechanism for selecting single arguments per argument cluster. We experiment with multiple methods and different models for selecting arguments. An overview of the methods used is given in Table 4.4. Below, we explain the setup for each method, and how we select the best-performing method to be used in the final output for HyEnA.

Prompts We construct different prompts for the two models to extract the desired argument selection output. *ChatGPT* is an instruction-tuned model and can be prompted to answer questions or follow instructions [293]. *Llama* lacks instruction-tuning, and thus requires prompts designed for next-token generation [387]. For the *ChatGPT* model, we instruct it with Prompt 1. For *Llama*, we use Prompt 2.

Prompt 1: ChatGPT

Consider the context of the COVID-19 pandemic and the following arguments:
- Argument 1
⋮
- Argument *k*

Write a key argument that summarizes the above arguments, and make it short and concise.

Prompt 2: Llama

Consider the context of the COVID-19 pandemic and the following arguments:
- Argument 1
⋮
- Argument *k*

A short and concise key argument that summarizes the above arguments is:

Testing Cluster Coherence First, we investigate the coherence of the clusters generated in Phase 2 according to each argument selection method, with the intent of measuring how each (automated) method aligns with the results of the first two phases of the (hybrid) HyEnA process. In cases of low coherence, semantically different arguments may end up together. Vice versa, in highly coherent clusters, only arguments that are the same are actually put together. While the error metric E (Equation 4.3) gives an error rate, it is mostly a *comparative* method, designed to select the best clustering method. Whether or not the clusters make sense to a human interpreter remains unclear. As such, we devise a so-called *odd-one-out* task, in which we use the Argument Selection methods for selecting arguments from a triple of arguments. In this triple, two arguments stem from the same cluster, and the third from a different cluster. The task for each argument selection method is to select which is the deviating argument. Here, we expect an adequate method to succeed well beyond random performance. Because Argument Quality is not intended for pairwise comparisons of arguments, we omit it in the odd-one-out task. We evaluate the remaining methods on a sample of 1K triples uniformly chosen from all possible triple combinations.

Evaluating Argument Selection We use different methods and different models for experimenting with the argument selection phase. As before, we employ an error metric to select the best-performing method, which we then inspect through a human evaluation. We use BERT score [449], a metric designed for model selection that uses a trained BERT model to compare the semantic similarity between the selected argument and the original opinions.

Specifically, BERT score recall correlates well with human *consistency* judgments, the factual alignment between selected argument and references (original opinions) [120]. We pick the best-performing method for argument selection based on this metric. This way, we penalize any possible effect of hallucinations of LLMs on the HyEnA method. We take the argument selected by each approach in the Key Argument Selection phase of the HyEnA procedure. As references, we take all comments that were involved in the creation of the cluster. We compute BERTScore and compare it across our approaches.

4.4.4 Baselines

We compare the output of HyEnA to the results of an automated and a manual approach to key argument extraction.

Comparison to Automated Baseline

We use the **ArgKP** argument matching model [33] 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 [144]. The original approach suggests relaxing heuristic parameters such that 20% of the opinions are selected as candidates. However, this caused overly specific arguments as candidates. Instead, we departed from the parameters used for the ArgKP dataset [33], and only relax them slightly such that ∼10% of opinions are selected as key point arguments.

Candidate key points and opinions are assigned a match score using a model trained for matching arguments based on RoBERTa [248]. Opinions only match the highest-scoring candidate key points if their match score exceeds a threshold θ, corresponding to the best match and threshold (BM+TH) approach. After deduplication, this results in a single list of key arguments per option. We use three metrics, *coverage* (C), *precision* (P), and *diversity* (D) to compare HyEnA and ArgKP.

Coverage (C) is defined as the fraction of opinions mapped to an argument out of all the processed opinions [33]. To compute C, first, we extract the set of key arguments $\mathcal{A}_H$ 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 $\mathcal{A}_A$ 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:

$$C_H = \frac{|O_H^{ann}|}{|O_H^{obs}|} \tag{4.4}$$

$$C_A = \frac{|O_A^{ann}|}{|O_A^{obs}|} \tag{4.5}$$

Comparing the coverage scores as defined above naively may not be fair since the set of observed opinions (i.e., the denominators of Equations 4.4 and 4.5) 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:

$$C_H^{common} = \frac{|O_H^{ann} \cap O_A^{obs}|}{|O_H^{obs} \cap O_A^{obs}|} \tag{4.6}$$

$$C_A^{common} = \frac{|O_A^{ann} \cap O_H^{obs}|}{|O_H^{obs} \cap O_A^{obs}|} \tag{4.7}$$

We add the same term to both denominator and numerator in Equations 4.6 and 4.7 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 [33]. 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 $\langle o_i, \mathcal{A}_H(o_i)\rangle$ and $\langle o_i, \mathcal{A}_A(o_i)\rangle$, where $\mathcal{A}_H(o_i)$ and $\mathcal{A}_A(o_i)$ 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:

$$P_H^{common} = \frac{\sum_{o_i \in O_H^{ann} \cap O_A^{ann}} z(o_i, \mathcal{A}_H(o_i))}{|O_H^{ann} \cap O_A^{ann}|} \tag{4.8}$$

$$P_A^{common} = \frac{\sum_{o_i \in O_H^{ann} \cap O_A^{ann}} z(o_i, \mathcal{A}_A(o_i))}{|O_H^{ann} \cap O_A^{ann}|} \tag{4.9}$$

Diversity (D) is defined as the ratio of key arguments and the number of comments seen by the method. We use diversity to signify how well our method is able to preserve the perspectives present in the opinions seen by the method. In order to compare across methods, we take (1) only correct mappings ($z(o_i, a_i) = 1$) using the labels from P and (2) take the opinions seen by both A and H. We define diversity as follows:

$$D_H = \frac{\mathcal{A}_H}{|O_H^{obs} \cap O_A^{obs}|} \tag{4.10}$$

$$D_A = \frac{\mathcal{A}_A}{|O_H^{obs} \cap O_A^{obs}|} \tag{4.11}$$

Comparison to Manual Baseline

A manual analysis involving six experts examined a portion of the feedback stemming from the PVE procedure. This analysis included a sample of participants (2,237 out of 26,293) for whom key arguments were identified [274]. Each expert generated a list of arguments for and against each of the relaxation measures based on the opinion text. A single participant could leave multiple opinions, and the analysis does not report the exact number of opinions analyzed. Since we have access to 36,781 opinions for the three options (Table 4.1), we estimate the number of opinions the six experts would have analyzed to be 3,129 across the three options (following each participant entering ± 1.4 opinions), and at least 2,237 (at

least one opinion per participant). In contrast, HyEnA annotators analyze 765 intelligently selected opinions across the three options.

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.

4.5 Results

First, we analyze the inter-rater reliability of annotations. Then, we analyze the intermediate results of the three phases of HyEnA. Finally, we compare our hybrid approach with the automated and manual baselines.

4.5.1 Annotator Agreement

Table 4.5 shows the inter-rater reliability (IRR) for four steps with overlapping human annotations. We didn't obtain IRR ratings for the argument extraction task in Phase 1 since the annotation is designed to be disjoint, and raters had little to no overlap in their extractions. In the Topic Generation phase (Section 4.1), we use the intraclass correlation coefficient $\mathrm{ICC}(3,k)$ [353] since it involves ordinal ratings. In the other three tasks, multiple binary labels are obtained for the same subjects. In these tasks, we use prevalence- and bias-adjusted κ (PABAK) [357], which adjusts Fleiss' κ for prevalence and bias resulting from small or skewed distribution of ratings.

In Topic Generation, the main source of the disagreement stems from a single option: REOPEN. Here, the annotators rated two topics almost inverted (rating 4 versus rating 2) out of a 1–5 Likert scale, resulting in an ICC score of 0.46. The two topics contained the words *"mental health income decrease,"* and *"measures rules these should"*. For the other two options, YOUNG and IMMUNE, a higher score of 0.71 and 0.80 were obtained respectively.

We obtained the lowest reliability scores for the last two annotation tasks, Key Argument Consolidation and Key Argument Evaluation. The obtained scores may be due to the difficulty of the task—for instance, lay annotators are asked to characterize the similarity between two arguments, and they may not stick to the provided definition of argument similarity. However, task difficulty may not be the only factor at play here. Argument comparisons are made with limited context, and the personal perspective or background of the annotator

Task	ICC3k	PABAK
Topic Generation	0.66 (0.14)	–
Topic Assignment	–	0.81 (0.10)
Key Argument Consolidation	–	0.34 (0.03)
Key Argument Evaluation	–	0.36 (0.04)

Table 4.5: IRR scores per task in HyEnA. We show the average (and standard deviation) over the three option corpora.

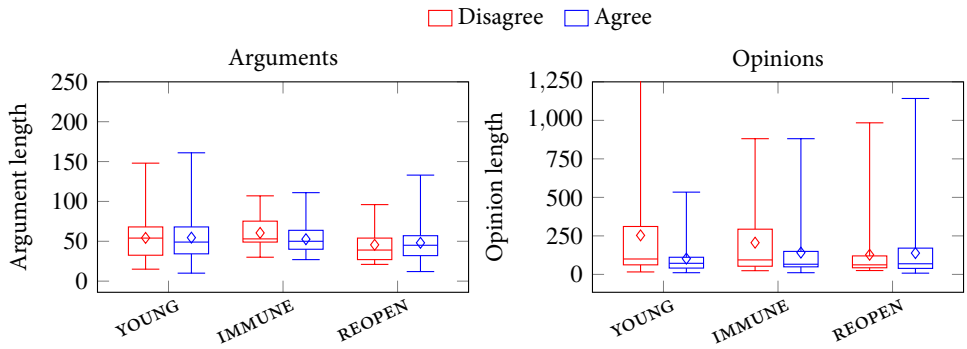

Figure 4.4: Disagreement analysis for the Key Argument Evaluation phase. On the left, argument lengths are the same whether annotators agree or disagree. However, on the right, annotators disagree on match labels in long opinions.

may influence their judgment. Thus, the low IRR scores may indicate a combination of task difficulty and the relatively subjective nature of the task [21]. Similar reasoning holds for the task of evaluating the match between the extracted argument and the original opinions.

Focusing on the evaluation phase, we compare argument–opinion pairs where large disagreement was observed (DISAGREE) to pairs with low disagreement (AGREE) in Figure 4.4. Specifically, we compared the lengths of the arguments and opinions. We find that the lengths of the arguments–opinion pairs with large inter-rater disagreement did not differ from those with low disagreement. However, we found considerably longer opinions on average when annotators disagreed. Possibly, long opinions contain multiple arguments, which in turn may cause the annotator to fail to identify the provided argument.

Prolific annotators were generally young (M=29.2, SD=7.8) and typically active users with a median of over 300 tasks completed (M=404, SD=418). A little over half of our annotators were male (58.8%), another 38.6% reported as female, and the rest had no data available. 76.7% reported a language other than English as their native language (we did require all annotators to be fluent in English). Annotators mostly resided in European countries, with the UK, Mexico, and the US being the only non-EU countries with more than 10 annotators. 23.8% reported as being a full-time student, with the rest either reporting as not being a student or having no data available. Further work is required in order to investigate the impact of demographic factors on the subjective interpretation of the opinions and arguments involved [352].

4.5.2 Phase 1: Key Argument Annotation

In Phase 1, individual annotators were guided through 51 opinions each and asked to annotate the observed arguments. Table 4.6 shows the number of different operations annotators perform over the 51 opinions. 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. Since the opinions had been automatically translated, we also provided annotators with the option to skip an opinion due to an unclear translation. Out of 51 actions, annotators reported mistranslations in 6, 7, and 2 opinions on average for YOUNG,

Option	Phase 1			Phase 2	
	# 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

Table 4.6: The average annotation operations (and their standard deviation) in Phase 1, and obtained statistics for Phase 2.

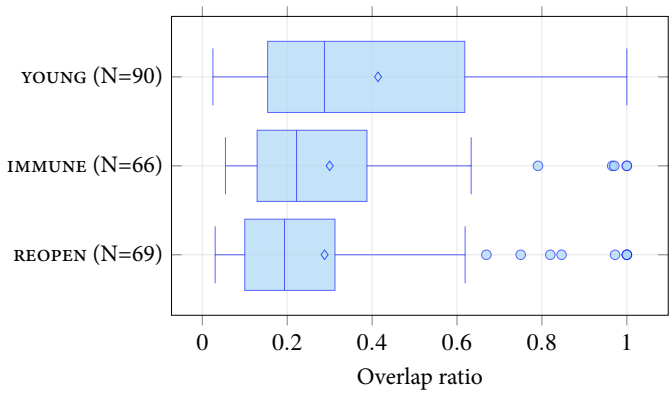

Figure 4.5: Distribution of argument overlap ratio for arguments generated by Key Argument Annotation in Phase 1.

IMMUNE, and REOPEN, respectively.

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.

Our instructions did not include an explicit mention of whether copying from the opinion text was allowed, but we observed that annotators often paraphrased arguments from opinions. To examine the behavior of the annotators, we measured the amount of text that was literally copied from the opinions. To do so, we take the largest common substring on the character level between opinion text and argument and divide it by the length of the argument. In Figure 4.5, we show the distribution of overlap ratios across all extracted arguments. While some arguments do get copied verbatim (overlap ratio of 1), across all three corpora annotators generally rephrase the arguments. This shows that, in HyEnA, human intervention acts in shaping the arguments extracted from the opinions, rather than simply copying part of an opinion (as automated methods would do). Table 4.7 shows some examples of arguments extracted with different overlap ratios.

The topic models for each option generated a large variety of topics. After the generation of the topic models $\mathcal{T}$, we retain only the top-15 most frequent topics to make the annotation

Option	Opinion Text	Extracted Argument	Overlap Ratio
YOUNG	Our daughter misses her friends so much and I notice that she really needs it	Positive for the psychological health of children	0.060
IMMUNE	Keep one system, keep it simple. Not too many deviations.	Everyone should be subject to the same set of rules/restrictions.	0.091
IMMUNE	Too little research has been done to limit the measures for people who are immune and too few opportunities to test it. In addition, it is difficult to control.	It is difficult to control.	1.000
REOPEN	These measures are quite easy to take compared to the unselected measures.	Measures are easy to take compared to the unselected measures	0.820

Table 4.7: Examples of extracted arguments in Phase 1 of HyEnA. Overlapping character sequences are highlighted in green.

Option	$\lvert\mathcal{T}\rvert$	Number of duplicates	Kept	Average rating
YOUNG	59	1	12	4.4
IMMUNE	56	2	12	4.4
REOPEN	72	0	14	4.0

Table 4.8: Expert topic generation statistics in Phase 1.

feasible. Our experts eliminated one, two, and zero topics as duplicates in the three options (Table 4.8). On average, the coherence scores—ranging from 1 (low) to 5 (high)—are high. This suggests that these topics were suitable for assignment to the arguments stemming from the crowd-extracted arguments. Table 4.9 shows examples from the final list of topics, with low-scoring topics removed.

4.5.3 Phase 2: Key Argument Consolidation

In Phase 2, HyEnA uses the POWER algorithm to guide human annotations on arguments similarity, with the intent of creating clusters of similar arguments across all arguments individually annotated in Phase 1. Table 4.6 (right side) shows the benefit of the POWER algorithm—the number of pairs requiring human annotation (Δ) was on average reduced by 60%. The transitivity scores τ [283] measure the extent to which transitivity holds among the similarity labels of argument pairs. The low τ scores indicate the need for subsequent clustering, given that there are no clear graph components in which all arguments are similar.

Figure 4.6 compares Louvain and spectral clustering for extracting argument clusters.

Option	Clarity Rating	Topic words
YOUNG	4.5	immune entertainment hospitality restrictions
	4	infection immunity risk infected
	4	*virus susceptible spread transmit*
	4.5	*schools reopen education students*
	5	*risk limited low dangerous*
	5	*group risk target least*
IMMUNE	4.5	homes nursing care vulnerable
	4	netherlands country provinces dutch
	5	*risk contamination danger dangerous*
	4.5	*work companies home economy*
	5	entertainment hospitality catering industry
REOPEN	5	homes nursing care vulnerable
	4	netherlands friesland groningen dutch
	5	risk hospitality entertainment dangerous
	3	*mental health income decrease*
	3	*measures rules these should*

Table 4.9: Examples of topics generated in Phase 1, including the top 4 words and the average clarity rating. Option-specific topics are *emphasized*.

Generally, both methods show a clear minimum for obtaining the final argument clusters. Louvain clustering yields the smallest error for the YOUNG and IMMUNE corpora, and spectral clustering for REOPEN corpus. These methods create 20, 14, and 18 clusters respectively. We pick these clusters as input to the argument selection phase.

Not all arguments inside the same cluster are constrained to have the same stance (pro or con) towards the policy option. We count what proportion of arguments in the cluster do not adhere to the majority stance. The distribution of stances scores is visualized in Figure 4.7. While we see that the upper limit is that half the arguments in each cluster are not agreeing with the majority label, the average ratio denotes that only a small fraction of argument stances do not agree with the majority stance label. This shows that the clusters generally represent a coherent distribution of arguments with similar stances to each policy option. The ratio on average is lowest for IMMUNE, which is the option with the highest ratio of con opinions.

4.5.4 Phase 3: Key Argument Selection

In Phase 3, we compare five Argument Selection methods for extracting a representative argument for each of the clusters obtained in Phase 2. We first perform an odd-one-out task to evaluate the coherence of the clusters according to each tested Argument Selection method (see Section 4.4.3 for additional details). Then, we evaluate the quality of the arguments that are selected to represent clusters.

Odd-one-out task Figure 4.8 shows the results of the odd-one-out evaluation. We perform pairwise statistical analysis by employing McNemar's test [101] with Holm-Bonferroni correction on multiple tests [4]. The test results indicate whether methods significantly differ

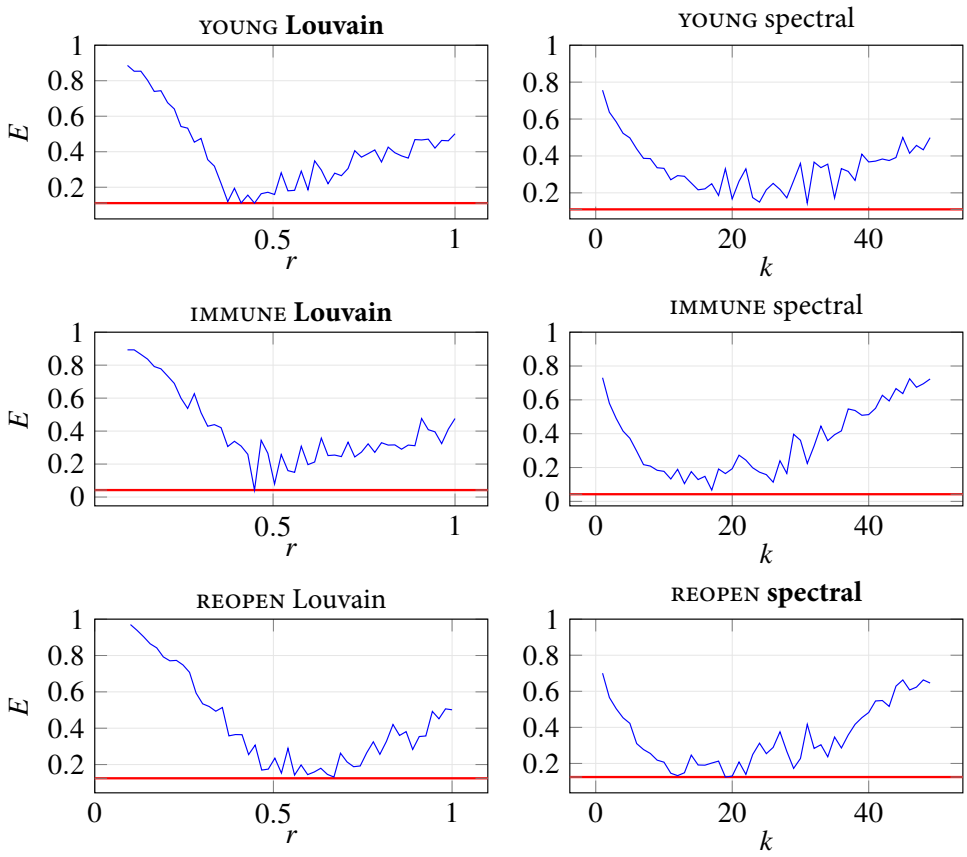

Figure 4.6: Error rate E for different parameters per clustering method (resolution parameter r for Louvain, k clusters for spectral) for each corpus in Phase 2.

in their misclassifications. We observe that only *Llama–random* does not have a significant difference in error proportions and can thus be assumed to perform similarly to each other. Conversely, two out of three methods outperform the random baseline. This indicates that these methods identify cluster membership relatively consistently with the results of HyEnA, although with considerable error rates. For Llama, we encountered a strong position bias with respect to the ordering of the triple: independently of which was the odd-one-out argument, the model primarily picks arguments at a specific index. This causes its performance to be similar to random picking. We attribute this to the lack of instruction tuning for the Llama model.

Evaluating Argument Selection To select the best-performing Argument Selection method, we compare BERTScores in Figure 4.9. We use the Kruskal-Wallis test (a non-parametric alternative to ANOVA since the scores are not normally distributed) to test whether all medians are equal at a 5% significance level [212]. Since we obtain a score well below our

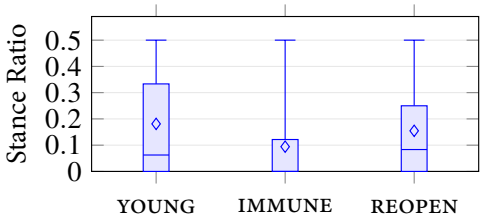

Figure 4.7: Stance distribution for clusters extracted for each corpus in Phase 2. A ratio of 0.5 denotes an equal number of pro and con arguments inside a cluster.

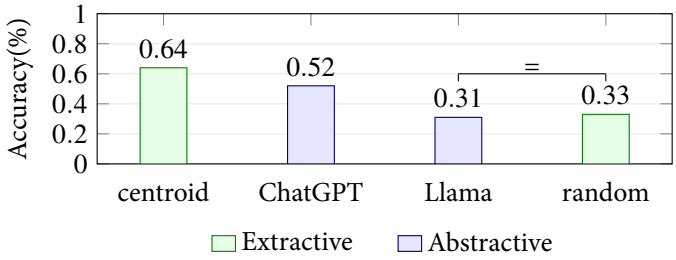

Figure 4.8: Accuracy on the odd-one-out task per method. Key Argument Selection methods marked with = do not significantly differ ($p < 0.05$) in their error proportions.

threshold, we conduct a post-hoc follow-up to identify pairs of significantly different Key Argument Selection methods. We employ Dunn's multiple comparisons of mean rank sums [107] with Holm-Bonferroni correction on multiple tests [4].

All extractive methods have a higher standard deviation than the generative methods. Some selected representative arguments likely caused the high maxima for extractive methods, since they are copied verbatim from opinions in the corpus. Conversely, the low minima are due to the extractive methods' inability to find representatives from the cluster (since there may be noisy clusters, see Figure 4.8). For the abstractive methods, the lower bound is higher, showing how rephrasing the selected argument makes it more related to all arguments inside a cluster. Between the abstractive methods, ChatGPT has a higher standard deviation than Llama. Since we did not perform extensive prompt engineering, there is room for improvement in both methods with better-crafted prompts.

The only significantly different method is Llama, with all others achieving similar BERT-Score performance. Surprisingly, none of the approaches on average performs considerably better than random. This suggests that selecting a representative argument from the cluster is relatively simple in practice because the argument clusters are sufficiently coherent. However, in the final evaluation, humans will be judging the match between selected arguments and individual opinions. Here, we strive for a better worst-case performance—we care less about having perfect matches, but rather wish to have fewer misrepresentation errors. Thus, given the comparable averages, we opt for the method with the highest lower boundary (the abstractive methods) and higher median score (ChatGPT outperforms Llama significantly), which we use for the remainder of the experiments.

Finally, we compare the output of Phase 3 of HyEnA against a version where the selection

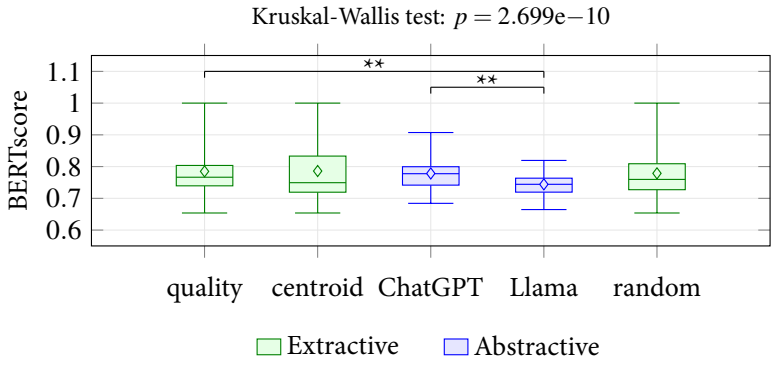

Figure 4.9: Aggregated BERTScore for the different Key Argument Selection methods across all corpora and argument clusters (Phase 3). Method pairs indicated by ** differ significantly from each other in median performance ($p < 0.05$).

Method	YOUNG	IMMUNE	REOPEN	**Overall**
HyEnA	0.816	0.833	0.641	0.765
HyEnA w/o Phase 3	0.787	0.848	0.739	0.789

Table 4.10: Comparing Precision (P) scores with and without Phase 3 (Key Argument Selection phase).

was manual. In particular, we take the extractions from Phase 1 and re-evaluate them using a new set of annotators. In Table 4.10, we show the difference in Precision (Equation 4.8).

We find that the addition of Argument Selection on average has a slight negative impact on the ability of annotators to match opinions and arguments. Most interestingly, when comparing argument matches for the same set of opinions before and after the addition of Argument Selection, we find that there is only fair agreement between the re-matched labels (Cohen $\kappa = 0.255$). This indicates that the argument selection phase makes annotating the match for some opinions to selected key arguments easier while making others more difficult. Selecting arguments using ChatGPT generates key arguments that are representative of the entire cluster, which can be more general than the arguments extracted by annotators from individual opinions. On the one hand, this can cause external annotators to not recognize the specific argument from a given opinion. On the other hand, it may result in annotators matching opinions and arguments on a more abstract level.

4.5.5 Comparison with Automated Baseline

Figure 4.10 compares the coverage, precision, and diversity scores 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 4.1)

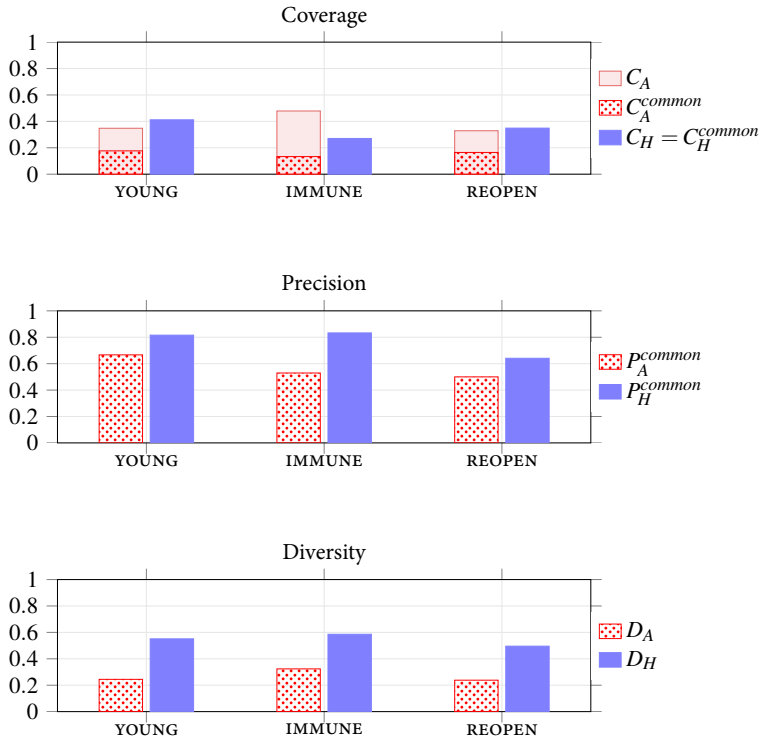

Figure 4.10: Comparing HyEnA and ArgKP.

was highly opposed and its corpus contains many repeated arguments. Since the set of *all* observed opinions is the entire corpus for ArgKP, the repeated arguments inflate its coverage. However, since HyEnA is designed to observe only a small subset of diverse opinions from the corpus, the repeated arguments do not influence its coverage significantly. This is corroborated in the diversity scores, where we observe HyEnA to consistently output a set of arguments that is more diverse than the ones produced by ArgKP.

In addition to comparing coverage over *observed* opinions, 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. ArgKP often fails to recognize the key arguments in the diverse set of opinions included by HyEnA.

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).

4.5.6 Comparison with Manual Baseline

Table 4.11 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

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

Table 4.11: A confusion matrix comparing the key argument lists generated by HyEnA and manual baseline. The complete mapping is given in Appendix C.3.

the lower human effort, the HyEnA lists largely overlap with the expert lists.

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 larger (random) sample of opinions experts analyzed did not include opinions supporting this argument, whereas the smaller (intelligently selected) set sampled in HyEnA did.

4.6 Discussion

We find that HyEnA exploits the strengths of automated methods and the insights from human annotation. HyEnA outperformed an automated KPA model in terms of precision and diversity, and on a diverse set of opinions, can capture more nuanced arguments. Further, HyEnA expanded beyond an expert analysis, showing how a fully manual procedure may also be limited. In the remainder of this section, we expand on three specific aspects.

Limitations Our experimental setup and comparisons are limited in their scope in multiple ways, thus making our conclusions hard to generalize. Our choice of baseline is the ArgKP model, which was optimized for the task of extracting Key Arguments from a corpus of opinions. However, other automated baselines are conceivable, especially with the introduction of the current generation of flexible LLMs (e.g., ChatGPT, Llama). Those models may be employed for KPA by using prompting techniques [245]. The capabilities of these models seem to imply that they have access to higher order argumentation knowledge [223], and thus would fare better than the basic ArgKP model. However, having such LLMs reliably process large amounts of citizen feedback without hallucinations is a nontrivial task, and the danger of models synthesizing ungrounded arguments exists [185]. In this process, due diligence to preserve a variety of perspectives is required (e.g., by optimizing for a range of opinions instead of single-annotator labels, Bakker et al. [28], Van Der Meer et al. [402]) in order to prevent rampant misrepresentation of marginalized demographics.

Instead of relying solely on the judgment of an LLM for the task of KPA, we opted to include one in the final step of HyEnA. While some of the criticism for using an LLM for end-to-end KPA still holds for the Argument Selection step as well, our method investigated a more controlled setup, supported by an objective task definition. Through our comparisons

with random and human-generated labels, we aim to show where, how, and to what extent LLMs may aid in the KPA process. As ever, the choice of metrics remains important for measuring the effect size.

Balancing Task Allocation 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, partly evidenced by the low IRR. While HyEnA reduced the number of comparisons required in the consolidation phase by 60%, we may experiment with different setups or other techniques for comparing arguments to remove this overhead. For example, first clustering the key arguments and then consolidating the arguments within these clusters (reverse order as HyEnA) may drastically reduce the number of judgments required in the second phase. Furthermore, future versions of HyEnA could benefit from investigating why annotators disagreed on labels in each phase, as it can lead to possible improvements in the annotation task.

We place human efforts in places where there are multiple bidirectional benefits possible stemming from performing the task. For instance, the Argument Annotation task both serves the purpose of analyzing the opinions to progress our method, as well as actively making annotators perform *perspective-taking*. On multiple occasions, annotators noted their increase in sympathy and recognition of the issues raised in the comments, showcasing how the task could further help bring understanding to a group of citizens.

Ablations studies All parts of the HyEnA pipeline are open to adjustment and can be performed by humans, machines, or a combination. In this work, we presented a specific version of this pipeline, but other ways of combining humans and AI are possible. However, the impact of choosing specific components remains unclear for parts of the pipeline, since we experimented with a single algorithm in some cases (e.g., the use of PageRank in Key Argument Consolidation, or the LLMs in Key Argument Selection).

HyEnA presents a general framework that allows individual phases to be supported by different types of technologies and different groups of crowd/expert annotators. Within this hybrid framework, we considered the following criteria when deciding to allocate tasks to humans or AI methods: (1) let humans read other's opinions to promote perspective-taking, (2) use humans to solve tasks where AI methods may incur considerable error, (3) leverage AI methods for routine tasks, and (4) use task-specific intrinsic evaluation metrics for selecting the right method.

In each phase, we perform both intrinsic evaluation (e.g., observe error rates for particular tasks or annotator behavior) and extrinsic evaluation against two baselines. This fits a standardized machine learning pipeline, except that we are now able to (1) evaluate annotator behavior and model performance jointly, and (2) make decisions on which techniques to use based on some intermediate statistic. We believe this setup to be generalizable for Hybrid Intelligence systems, as it makes the role of the designer and their decisions explicit [5]. Furthermore, the results remain interpretable, as any decision made by either annotators or models can be traced from opinion to selected key argument.

Different configurations of the HyEnA framework are possible, and the one we have presented is an instance that tackles the problem of policy feedback analysis. HyEnA is a complex combination of AI methods and human annotation. Our main objective was to

present the HyEnA framework, as well as a real-world use case to show the benefit of using a Hybrid Intelligent methodology. However, other choices for individual components of HyEnA can be used, or parts of the method can be performed solely by humans or AI methods. We leave this open for future work, as swapping out components is not straightforward and requires considerable amounts of work. We envision research to come up with similar use cases where HI can make a significant impact.

4.7 Conclusion and Future Directions

We develop and evaluate HyEnA, a hybrid method that combines human judgments 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.

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 [238, 400], sentiment, emotion, and attribution [411]. 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.

Ethical Considerations

This chapter develops and evaluates a hybrid (human and AI) approach to extracting key arguments from an opinion corpus. The intended use case for our method is synthesizing key arguments that are grounded in opinionated policy-related comments, by using a pool of annotators. We identify two main aspects of risk in our method.

First, we aim to mitigate the effect of individual biases by grounding the key arguments in general public user opinions. However, the key argument extraction is ultimately performed by individual annotators. We address the influence of subjectivity and noise by combining multiple annotators in the consolidation phase. Further, as our method is transparent, the complete annotation process (from opinions to consolidated key arguments) is traceable. One could implement additional checks on annotator behavior as a bias-mitigating factor, which is a significant research challenge on its own.

Second, the diversity of the opinion embeddings is contingent on the representational quality of the S-BERT model. Underlying biases in its representation may influence the opinions sampled. However, we use FFT to actively sample diverse opinions, which can reduce the impact of inaccurate embeddings.