# Rebuta: Automated Debate Argument Evaluation

**Rahul Bachal**
Department of Computing and
Mathematical Sciences
Caltech
Pasadena, CA 91125
rbachal@caltech.edu

**Justin Leong**
Department of Computing and
Mathematical Sciences
Caltech
Pasadena, CA 91125
jleong@caltech.edu

**Christina Lin**
Department of Computing and
Mathematical Sciences
Caltech
Pasadena, CA 91125
cylin@caltech.edu

**David Qu**
Department of Computing and
Mathematical Sciences
Caltech
Pasadena, CA 91125
dqu@caltech.edu

## Abstract

In this paper, we propose a new algorithm for scoring debate arguments. By adding novel features such as transitional phrase types and syllable count, and by systematically testing feature combinations, our algorithm outperforms the state-of-the-art essay scoring algorithm and the state-of-the-art argument scoring algorithm. The final model is applied in Rebuta, an online debate platform that scores user-contributed arguments and displays the arguments for each side sorted by score, allowing users to read the best arguments first.

## 1 Introduction

### 1.1 Motivation

Debates on popular sites such as Facebook, YouTube, and Twitter, often degenerate into name-calling and trolling without offering a productive space to resolve difficult questions. Questions are met with inconclusive responses, even on specialized question-answer sites such as Quora and Stack Overflow. With increasing quantity of content, it is difficult for people to engage in productive, collaborative conversations and debates on the internet without getting bogged down in spammy, irrelevant noise.

### 1.2 Overview of Work Done

Our major contributions are two-fold as follows:

(1) Using a combination of syntactic and content features, we have developed a machine learning algorithm to automate evaluation of debate arguments. Our algorithm outperforms the state-of-the-art style-based algorithm[9] by 31.06% and the state-of-the-art content-based algorithm[10] by 6.54% on the labeled debate dataset.

(2) We have created the application framework for Rebuta, an online debate platform that would make use of this argument scoring algorithm. Any user-contributed argument receives a score on a scale of 1-10 in the context of the debate topic. By providing immediate feedback, the platform encourages and incentivizes users to contribute high-quality content. We hope to build a meaningful space for discussion that caters to the needs of its users.

## 2 Related Work

### 2.1 Grading Writing Style in Essays

In 2012, the Hewlett Foundation held a Kaggle competition on automated essay scoring. The foundation provided a set of around 13000 essays with scores. One of the best performing algorithms was developed by Manvi Mahana, Mishel Johns, and Ashwin Apte of Stanford University[9]. Their approach featurized word count, sentence count, part of speech counts, comma count, punctuation count, and quotation mark count. A forward feature selection algorithm was used to choose the best combination of features, and predictions were made with a linear regression model. Their model achieved a kappa score of 0.703 on the essay set and 0.531 on the debate set. Other similar, high-performing algorithms that focused featurizing words and grammar include Shihui Song and Jason Zhao's submission[12]. As acknowledged by Mahana, Johns, and Apte, these features capture the writing style and syntax well but fail to judge essays in which the context is important.

### 2.2 Grading Argument Strength in Student Essays

A quality contribution to a debate should be not only well-written but offer a strong argument. The latest work in the field of modeling argument strength comes from Isaac Persing and Vincent Ng in 2015[10]. Persing and Ng's approach uses features such as part of speech N-grams, semantic frames, transitional phrases, coreference, prompt agreement, argument component predictions, and argument errors. They developed an argument corpus of essays on 10 prompts annotated with numerical scores. However, this corpus was not free for download, so for the purpose of comparing performances, we applied their algorithm to our essay and debate sets. Persing and Ng's algorithm achieved kappa scores of 0.701 and 0.654 on the essay and debate sets respectively.

Jill Burnstein and Daniel Marcu's research on identifying theses and conclusions in student essays[5] is also relevant to evaluating . The presence of a strong thesis and conclusion suggests good structure in an argumentative piece of writing. [13]

### 2.3 Sentiment Analysis of Social Media Content

Unlike essays and articles, the writing found on social media websites tends to be informal, short, and often charged with emotion. Research on sentiment analysis of Twitter, a platform dedicated to concise 140-character messages, tackles the issue of deducing emotion from social media content. Song Feng et al analyzed a corpus of tweets classified as positive or negative sentiment by corresponding emoticons[7]. Their model was tailored to the particularities of social media content.

### 2.4 Analysis of Ideological Online Debates

Swapna Somasundaran and Janyce Wiebe evaluated the sentiment and arguing opinions to identify stances in ideological debates [11][6]. They restricted their data set to argumentative writing and tailored featurization to target key words and phrases that are commonly found in written debates. They established connections between certain words and indications of argument, sentiment, and the stance of the writer on the debate.
A key part of Somasundaran and Wiebe's work was identifying and collecting product and political debate data. As our works overlap in areas of application, we appreciate the political debate set that they provided.

### 2.5 Debate Platform Competitors

Existing online debate platforms include

1. debate.org - Back in 2007, debate.org was released as a tool for debating. It was acquired in 2010 for 15x ROI (according to the founder's LinkedIn), and has a respectable userbase of 300,000 cumulative users. The site can not really be considered popular or mainstream, and the technology and interface are out-of-date.[3]
2. createdebate.com - Also released around the 2008 election, Create Debate is another debating site that has a decent userbase with just less than 100,000 total users. Like debate.org, the technology and interface are dated.[1]

3. Moot app - A more recent attempt is the Moot iOS App, which gives users a way to post opinions on different sides, but doesn't allow meaningful interactions between users in building arguments or counterarguments. [2]
4. slant.co - A website originally targeted toward the tech community, slant.co has grown in the past few months to include fashion and other topics. Slant allows users to add "pros" and "cons" to arguments and to upvote other users' contributions. Our original approach was to create a website with an interface similar to slant.co but we have adapted our approach since slant.co has rapidly developed to fill the niche in the market.[4]

## 3 Algorithm Design

### 3.1 Overview

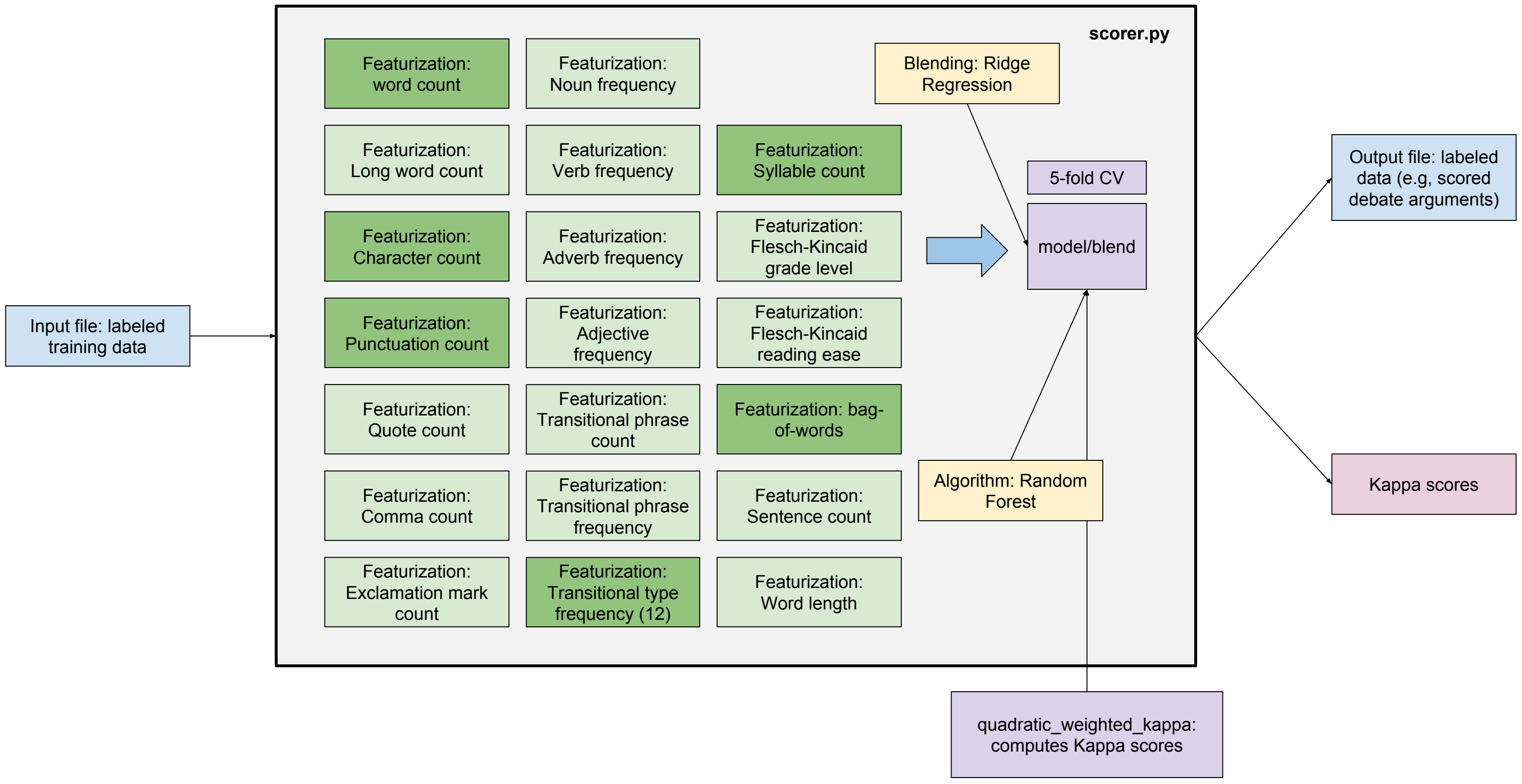
The objective of our algorithm is to provide a ranking similar to that of a human grader based on how useful an argument is to a debate. We measure performance using kappa score which is a measure of inter-grader agreement on a particular set of data. This was the metric used in the Kaggle essay scoring competition to gauge the accuracy of the scoring algorithms.

In addition, we used 5-fold cross validation for calculating the kappa scores for each model to prevent overfitting. Using 5-fold cross validation as opposed to just a singular training and test set was important because we would be doing an exhaustive search using a subset of the features. By using 5-fold cross validation, the more generalized models will do better and thus reduce the chance of statistical flukes (e.g. a model performs abnormally well because when sampling thousands of models, there is likely to be events performing several standard deviations better just by chance).

### 3.2 Features

The full list of features tested are as follows:

1. Word count
2. Long word count
3. Noun frequency
4. Verb frequency
5. Comma frequency
6. Adjective frequency
7. Adverb frequency
8. Character count
9. Punctuation count
10. Quotation marks
11. Sentence count
12. Word length
13. Exclamation marks
14. **Syllable count**
15. **Flesch-Kincaid Reading Ease** - higher scores indicate material that is easier to read; lower numbers mark passages that are more difficult to read
16. **Flesch-Kincaid Grade Level** - same as above except scaled to present score as a grade level
17. Transitional Phrase count
18. Transitional Phrase frequency
19. Bag of Words - trained on the most common words from all the training data
20. **Transitional Phrase type frequency - 12 different types (each one used as a feature) (e.g. addition, consequence, diversion, etc.)**



scorer.py
Featurization: word count
Featurization: Noun frequency
Blending: Ridge Regression
Featurization: Long word count
Featurization: Verb frequency
Featurization: Syllable count
5-fold CV
Output file: labeled data (e.g, scored debate arguments)
Featurization: Character count
Featurization: Adverb frequency
Featurization: Flesch-Kincaid grade level
model/blend
Input file: labeled training data
Featurization: Punctuation count
Featurization: Adjective frequency
Featurization: Flesch-Kincaid reading ease
Featurization: Quote count
Featurization: Transitional phrase count
Featurization: bag-of-words
Kappa scores
Featurization: Comma count
Featurization: Transitional phrase frequency
Featurization: Sentence count
Algorithm: Random Forest
Featurization: Exclamation mark count
Featurization: Transitional type frequency (12)
Featurization: Word length
quadratic_weighted_kappa: computes Kappa scores

Figure 1: The algorithm is implemented as scorer.py, which encompasses 20 features. The darker green blocks represent features that were selected for the final model. Only one transitional phrase type, diversion, was selected.

The first 16 features are based on using syntax and writing style while the last 4 features are based on scoring the argument. Bolded features were the novel features that our team added. Most of the syntactic features were from the Kaggle competition and most of the argument scoring features are from Persing and Ng's study on argument strength[10].

### 3.3 Methodology

With 31 features, there are 300,540,195 to choose 15 features. Thus, there are restrictively many(over a billion) possible combinations of features to test. While each individual combination of features took several hours to test on our computers, this was reduced to a matter of minutes on a server with code optimized to reuse previous computations. To select features, we followed a methodology similar to that of Mahana, Johns, and Apte[9]. Our algorithm used a script to add one feature at a time and check if the kappa score improved. If it did improve the model, it was added to a list of potential features. After running this script several times with different randomizations, we were able to select all 12 possible contributing features to the best model (word count, character count, punctuation count, quotation marks, sentence count, syllable count, bag of words, addition, consequence, contrast, direction , and diversion transitional phrase types).

After narrowing down the list to 12 features, we used another script to iterate through all possible combinations of these remaining features. The same bag of words model was blended each time to reduce variation and computation time. From this exhaustive search, we determined that the best performing model used the word count, character count, punctuation count, syllable count, diversion transitional phrase type, and the bag of words model.

## 3.4 Data

We tested on two data sets because the essay data set was readily available with scores by expert graders. This also enabled us to compare against the Kaggle competition winners in essay scoring. After testing on this initial data set, we labeled a set of debate arguments on gun control and measured the algorithms' performance on these arguments.

### 3.4.1 Scored Essay Set

For this data set, there are eight essay sets. Each of the sets of essays was generated from a single prompt. Selected essays range from an average length of 150 to 550 words per response. Some of the essays are dependent upon source information and others are not. All responses were written by students ranging in grade levels from Grade 7 to Grade 10. All essays were hand graded and were double-scored. Each of the eight data sets has its own unique characteristics. The variability is intended to test the limits of our scoring engine's capabilities.

### 3.4.2 Scored Debate Set

We downloaded a data set from Dr. Swapna Somasundaran with 1069 posts made by users of an online forum. Each post was a response to a debate on gun control. The data was not labeled with scores for argument quality, so we labeled the posts using a specific grading rubric. The first 50 posts were labeled by three graders in order to calibrate the rubric. The inter-rater agreement with the ground truth, which was the average of the three graders' scores, for these posts was 0.684.

| Score | Expectations |
|---|---|
| 1 | Troll comments. Statement(s) of useless facts only. Multiple unrelated argumentative sentences with no explanation. |
| 2 | Statement(s) of facts only which implies an argument. Unsuccessful attempt at constructing argument: multiple related statements or arguments that do not really support one another. |
| 3 | Attempt at an argument with supporting sentences/explanations, but not convincing due to weak logic, lack of evidence, etc. |
| 4 | Argument with good but incomplete support (for instance, making a two-part claim but only defending one part). Good support = explanations, quotes. Or, semantically complete but incohesive argument. |
| 5 | Argument with close to adequate support. |
| 6 | Argument with adequate support for all aspects of the argument. |
| 7 | Complex argument(s) with adequate support. |
| 8 | More complex argument, with convincing supporting statements that address opposing viewpoints. Or, there are multiple points that support the overarching argument being made. |
| 9 | Contains all the properties of the previous score in addition to being exceptionally well written and well supported. |
| 10 | Outstandingly well-written, complex, fully developed, fully supported argument. |

## 3.5 Performance

To measure performance of our model, we would compare to two baselines: the model which resulted from the results of the Kaggle essay scoring competition[9] and the model from Persing and Ng on argument strength[10]. For the model from the Kaggle competition, the authors took a systematic approach to adding features by starting with the best performing feature and adding them one at a time and verifying whether or not each feature improved the previous model. To replicate their model on the debate data, we went through the same process with our automated script to determine the baseline model. The features chosen by this process were character count, word count, nouns, verbs, adjectives, adverbs, comma count, punctuation count and quotation count. The overall kappa score was $0.531415$ which was lower than the single feature bag-of-words kappa score.

Next to replicate Persing and Ng's model as a baseline, we combined bag-of-words, transitional phrases frequency, and parts-of speech frequencies. They also had coreference and prompt agreement as features. While we did implement coreference as a feature, we could not use either of these in our framework since we were scoring debate arguments instead of essays and there were no prompts or reference labels so we could not measure coreference or prompt agreement. Although their paper originally used a SVM, we used a Random Forest Regressor instead because the kappa score was better ($> .05$) than when a SVM was used. This baseline far outperformed the essay scoring baseline with a kappa score of $0.65375$.

At first, because we were computationally constrained to our computers, we replicated the systematic approach to adding features from the paper based on the Kaggle competition. This led to the best score of $0.65877$ which was only $0.77\%$ better than the state-of-the-art argument scoring baseline. The model used the features character count, word count, nouns, verbs, adjectives, adverbs, comma count, punctuation count, quotation count, transitional phrase frequency, bag of words. Once we had access to a high performance computing server, we could run this approach with different parameters to the Random Forest Regressor in order to find the relevant features as described in 3.3. By automating the testing process and adding in the transitional phrase types, the kappa scored improved over $6\%$ to the current best kappa score of $0.6964957006$. Since this kappa score was greater than that of the average trained human scorer, the potential for improvement is limited by the accuracy of the ground truth labels. By crowdsourcing the debate scoring or by interactively teaching the crowd to label the data[8], we may be able to achieve better ground truth labels.

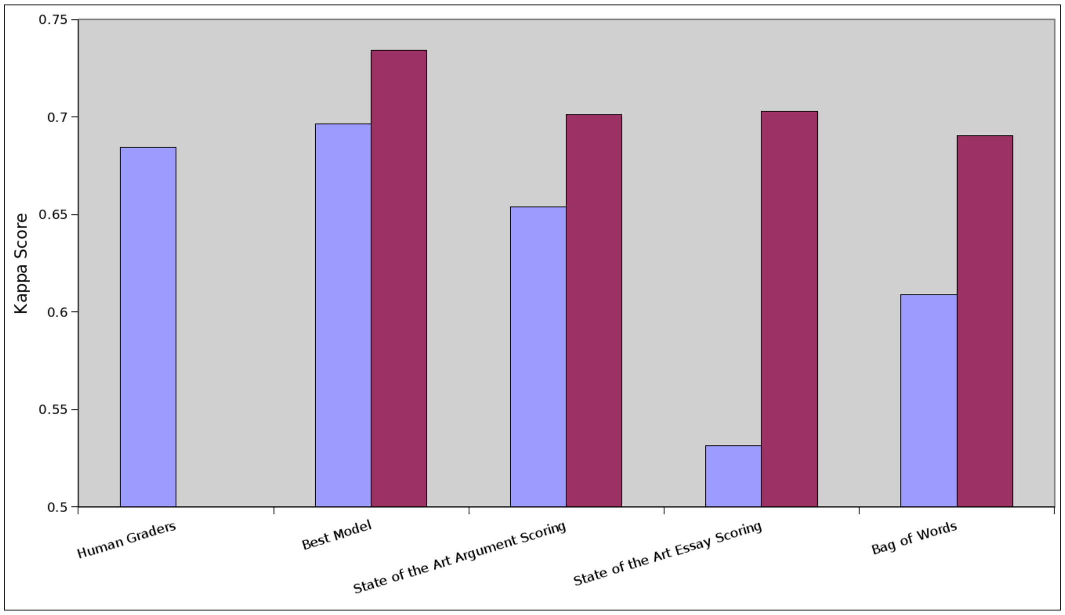


Figure 2: The model we developed perform better than the state of the art algorithms as well as outperforming human graders in terms of accuracy. The "Best Model" had over a 6.54% improvement over the "State of the Art Argument Scoring" algorithm.

|  | Kappa Scores | Standard Deviation |
|---|---|---|
| Human Graders | 0.684 | 0.172 |
| Best Model | 0.697 | 0.023 |
| State of the Art Argument Scoring | 0.653 | 0.028 |
| State of the Art Essay Scoring | 0.531 | 0.039 |
| Bag of Words | 0.609 | 0.027 |
| Word Count | 0.327 | 0.048 |
| Character Count | 0.322 | 0.075 |

## 4 App Design

`http://59c9f39c.ngrok.io`

### 4.1 Overview

The Rebuta app is designed primarily with a website interface. We identify three modules: scoring, auth, and debate. As discussed in the Algorithm Design, the scoring module consists of the scoring algorithm and scores debate arguments on a scale of 1-10. The auth module includes user authentication. We use this system to authenticate users and provide security. In addition, we identify authors publicly to encourage more genuine content. Finally, we have the debate module, which includes all functionality for working with debate data. We describe the auth and the debate modules in further detail.

### 4.2 User Authentication Module with Facebook

Our authentication is built using the OAuth standard. This framework works by asking a third-party identity and API provider to authenticate users. This provider then returns a key providing access to the core APIs for a user. This is distinct from OpenID, another authentication framework. OpenID provides a certificate from a third-party identity provider instead of a key. We use Facebook to provide our authentication. This has the benefit in that users are identified on Rebuta with their real names. The OAuth framework is implemented using facebook-sdk. User data is stored in a MySQL database. We also provide a front-end UI that allows users to login, logout, and view their profile information.

### 4.3 Debate Module

The debate module supports creation and viewing of debate data on the Rebuta website. The debate data is stored in a MySQL database. See the Appendix for a detailed schema of the database. The Rebuta website directs users to a dashboard, which shows all debates as well as their positions and the scores of each position. Clicking on a debate shows the posts for each position in the debate. The user then has the option to upvote or downvote arguments as well as add a new argument to the debate. See the screenshots in the Appendix for an illustration.
The website runs on Python Flask with a MySQL backend. The webpage is rendered using the Jinja2 templating language with HTML. Each endpoint on the website routes to a specific template as well as reads or writes from the database. For example, /debate/1 will route to the debate with id 1 in the database. These requests are all performed using GET. We use POST requests to submit data from forms, including creating a new debate and adding a post.

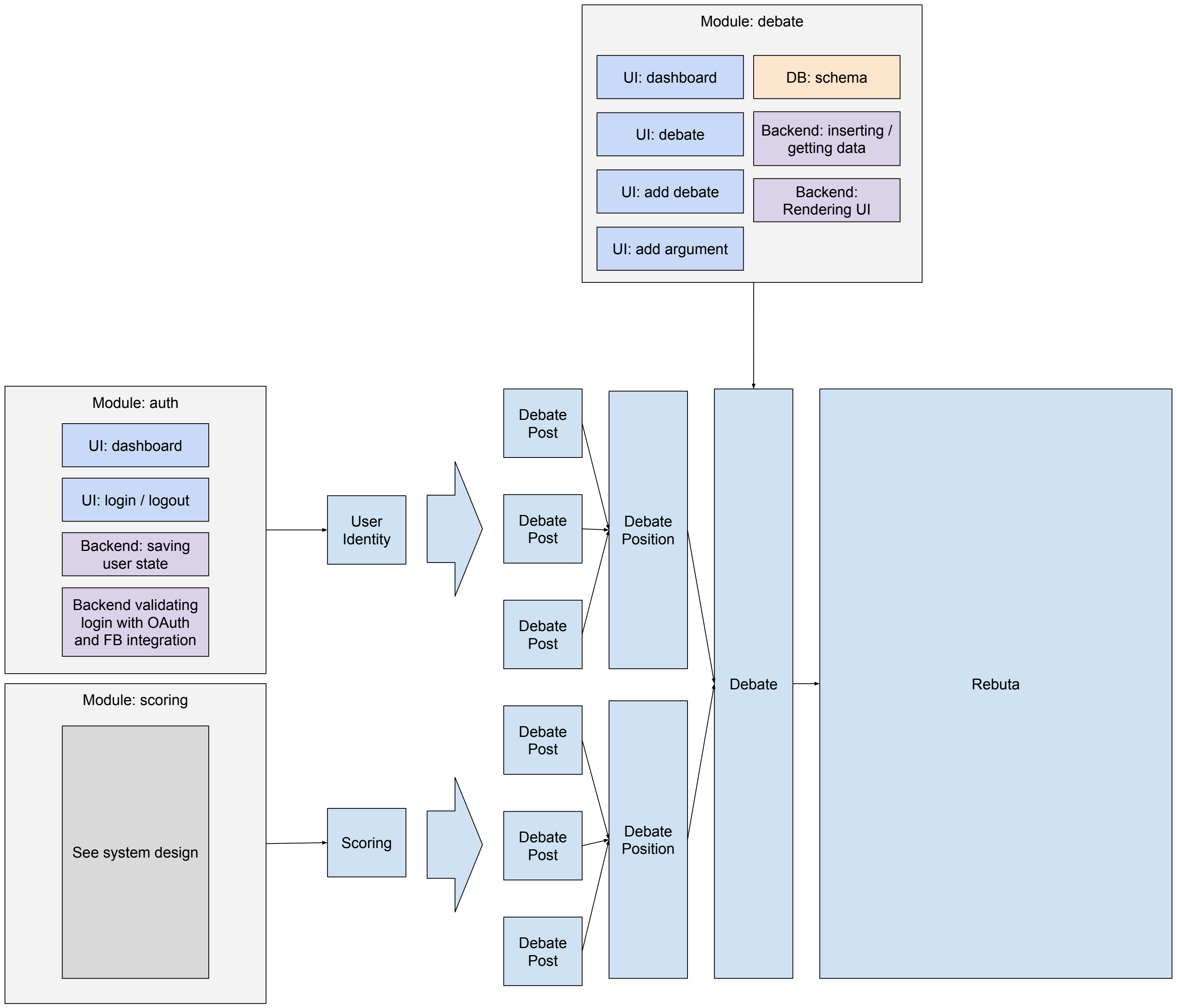


Figure 3: The three backend components of the Rebuta website are the scoring, auth, and debate modules. The scoring module is the final argument scoring model built by out algorithm, while the auth and debate modules allow users to log in, create debates, and add arguments.

## 5 Future work

### 5.1 Addressing the Cold Start Problem in New Debate Topics

Bag-of-words featurization is a major component of our algorithm: using bag-of-words as the lone feature yields a kappa score of 0.691 on the essay set and 0.609 on the debate set. However, this feature requires existing essays and debate arguments on the same topic to build a dictionary and train the model. When a new debate topic is created on Rebuta, there will not be existing arguments to train on. We would like to explore the possibility of using news articles from reputable sources to aid in bag-of-words featurization. For most debate topics that would appear on a social media platform, there are likely to be existing news articles that are relevant.

### 5.2 Using the Scoring Algorithm on Comments

Bloggers and website moderators often have the overwhelming task of going through dozens, hundreds, or even thousands of comments and often the first comment is most easily viewed instead of the most relevant comments. While having an upvote system solves the relevancy problem, hateful or derogatory statements may be upvoted instead of useful comments. Our algorithm could be applied to a blog tool to solve this problem. In addition, our algorithm is easily adaptable such that by scoring several comments, moderators can teach the algorithm what writing style or set of words is preferable. Thus, the moderators' job becomes easier since there would be more automation involved. It would be interesting to create a module using our algorithm which uses this approach.

### 5.3 Improving the Website

The current Rebuta website is static and allows user interaction only in terms of redirecting to other pages. For example, when a user wants to create a new post, we redirect the user to a different endpoint on the website. We would like to add JavaScript and JQuery to the website to allow dynamic user interaction. This will also enable features such as upvoting and downvoting as well as commenting on posts.

## 6 Conclusion

We have developed a new algorithm for scoring debate arguments that incorporates syntactic features (word count, character count, punctuation count, syllable count) as well as content features (bag-of-words, diversion type transitional phrases) and uses a random forest to produce a final model. Using the Kappa score metric, we have shown that our algorithm outperforms the state-of-the-art style-based algorithm[9] by 31.06% and the state-of-the-art content-based algorithm[10] by 6.54%. Our algorithm performs remarkably well on the gun control training data, and we intend to confirm the results on other debate data in the future.

We have also created a product that facilitates online debates by automatically scoring debate posts and providing an online interface for users to add content. The Rebuta website at its current stage is an minimum viable product for debating online. Users are able to create debates with multiple positions, contribute to a position in a particular debate, and receive scores for posts from our algorithm. Each position for a particular debate also has a total score, allowing users to quickly determine which position is dominant. The website encourages users to contribute high-quality content by providing immediate feedback on user posts as well as associating users with their real names from Facebook.

**Acknowledgments**

We would like to thank Professor Farshad Lahouti for his guidance and mentorship as well as Zilong Chen and Kaikai Sheng for their input and advice. We would also like to thank Jordan Bonilla for use of his server for the computational power to make our analysis possible.

## References

[1] Debate forum | online debate community | createdebate. `http://www.createdebate.com/`.

[2] Moot - social debate platform. `https://itunes.apple.com/us/app/moot-social-debate-platform/id912267008?mt=8`.

[3] The premier online debate website | debate.org. `http://www.debate.org/`.

[4] Slant - home. `http://www.slant.co/`.

[5] Jill Burnstein and Daniel Marcu. A machine learning approach for identification of thesis and conclusion statements in student essays. *Computers and the Humanities*, 37(4):455–467.

[6] Noura Farra, Swapna Somasundaran, and Jill Burnstein. Scoring persuasive essays using opinions and their targets. In *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*, page 64–74.

[7] Song Feng, Jun Seok Kang, Polina Kuznetsova, and Choi Yejin. Connotation lexicon: A dash of sentiment beneath the surface meaning. *ACL*, pages 1774–1784, 2013.

[8] Edward Johns, Oisin Mac Aodha, and Gabriel J. Brostow. Becoming the expert - interactive multi-class machine teaching. *CoRR*, abs/1504.07575, 2015.

[9] Manvi Mahana, Mishel Johns, and Ashwin Apte. Automated essay grading using machine learning. 2012.

[10] Isaac Persing and Vincent Ng. Modeling argument strength in student essays. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, page 543–552.

[11] Swapna Somasundaran and Janyce Wiebe. Recognizing stances in ideological on-line debates. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, page 116–124.

[12] Shihui Song and Jason Zhao. Automated essay scoring using machine learning. 2013.

[13] Henning Wachsmuth. 3.23 analysis of stance and argumentation quality. *Debating Technologies*, page 37.

## 7 Appendix

### 7.1 Gitlab Repositories

Model Training and Algorithm:

`https://gitlab.com/midnightbedlamites/automated-essay-scoring.git`

Website: `https://gitlab.com/alphaz99/rebuta.git`

### 7.2 Model Modules

| Module Name | Description/ Function | Authorship | Technology used | Workload | Other issues (e.g. complexity) |
|---|---|---|---|---|---|
| Automated Testing and Cross Validation | Model Training | Justin Leong, Rahul Bachal, Christina Lin | Python, sklearn, numpy, pickle, itertools | 220 lines | exhaustive search (also had versions for greedy search that were not included) |
| Model Training, Blending, and Scoring | Model Training | Rahul Bachal, Justin Leong | Python, sklearn, numpy, pickle | 200 lines | used for testing individual features and hard coded combinations of features |
| Bag of Words | Feature | Rahul Bachal | Python, sklearn | 50 lines | Long time to train model - just included same bag of words model and blended it with other features for efficient testing |
| Word, Character, Comma, Quote, Punctuation, and Sentence Counts | Features | Justin Leong, Christina Lin | Python | 40 lines | O(n) for each feature where n is the size of the input for feature extraction |
| Parts of Speech Frequency | Features | Justin Leong | Python, nltk | 80 lines | Extremely slow to label individual parts of speech - did not improve model |
| Flesch-Kincaid Reading Level, Reading Ease, Syllable Count | Feature | Christina Lin | Python, nltk, nltk.corpus. cmudict | 80 lines | N/A |
| Transitional Phrases (by type) | Features | Justin Leong | Python | 100 lines | fast but many features (12 different types) |
| Kappa Scoring | Scores Inter-rater agreement | Kaggle Competition, Justin Leong | Python, Matlab, R | 300 lines | implemented by contest judges |

### 7.3 Website Modules

| Module Name | Description/ Function | Authorship | Technology used | Workload (lines) | Other issues (e.g. complexity) |
|---|---|---|---|---|---|
| Auth | User Authentication and Security | Rahul Bachal, David Qu, Justin Leong, Christina Lin | OAuth, facebook-sdk | 100 lines | Integration with Facebook App |
| Dashboard Frontend | UI design, UI implementation | Christina Lin | Flask, Bootstrap | 50 lines | UI design |
| Debate Frontend | UI design, UI implementation, frontend integration | Rahul Bachal, Christina Lin | Flask, Bootstrap | 200 lines | UI design, using Jinja2 templating to integrate HTML with Flask |
| Debate Backend | Backend for Debates | Rahul Bachal, Christina Lin | Flask, MySQL | 200 lines | Flask helper functions to access data |
| Users | DB Table: users | Rahul Bachal | MySQL | 12 lines | N/A |
| Debates | DB Table: debates | Rahul Bachal | MySQL | 5 lines | N/A |
| Debate Positions | DB Table: debate_positions | Rahul Bachal | MySQL | 8 lines | N/A |
| Debate Posts | DB Table: posts | Rahul Bachal | MySQL | 9 lines | N/A |

### 7.4 DB Schema

```
DROP TABLE IF EXISTS posts;
DROP TABLE IF EXISTS users;
DROP TABLE IF EXISTS debate_positions;
DROP TABLE IF EXISTS debates;

CREATE TABLE users (
    user_id INTEGER NOT NULL AUTO_INCREMENT,
    auth_id VARCHAR(255) NOT NULL, -- Facebook identifier
    first_name VARCHAR(255) NOT NULL,
    last_name VARCHAR(255) NOT NULL,
    created DATETIME NOT NULL,
    updated DATETIME NOT NULL,
    profile_url VARCHAR(255) NOT NULL,
    access_token VARCHAR(255) NOT NULL,
    PRIMARY KEY (user_id),
    UNIQUE (auth_id)
);

CREATE TABLE debates (
    debate_id INTEGER NOT NULL AUTO_INCREMENT,
    debate_name VARCHAR(255) NOT NULL,
    PRIMARY KEY (debate_id)
);

CREATE TABLE debate_positions (
    position_id INTEGER NOT NULL AUTO_INCREMENT,
    debate_id INTEGER NOT NULL,
    position_name VARCHAR(255) NOT NULL,
    position_description TEXT NOT NULL,
    PRIMARY KEY (position_id),
    FOREIGN KEY (debate_id) REFERENCES debates(debate_id)
);

CREATE TABLE posts (
    post_id INTEGER NOT NULL AUTO_INCREMENT,
    debate_id INTEGER NOT NULL,
    position_id INTEGER NOT NULL,
    post_text TEXT NOT NULL,
    post_author INTEGER NOT NULL,
    post_score INTEGER NOT NULL,
    PRIMARY KEY (post_id),
    FOREIGN KEY (debate_id) REFERENCES debates(debate_id),
    FOREIGN KEY (position_id) REFERENCES debate_positions(position_id),
    FOREIGN KEY (post_author) REFERENCES users(user_id)
);

CREATE VIEW debate_position_scores AS
    SELECT debate_id, position_id, SUM(post_score) AS total_score
        FROM posts
        GROUP BY debate_id, position_id;
```

## 7.5 Screenshots

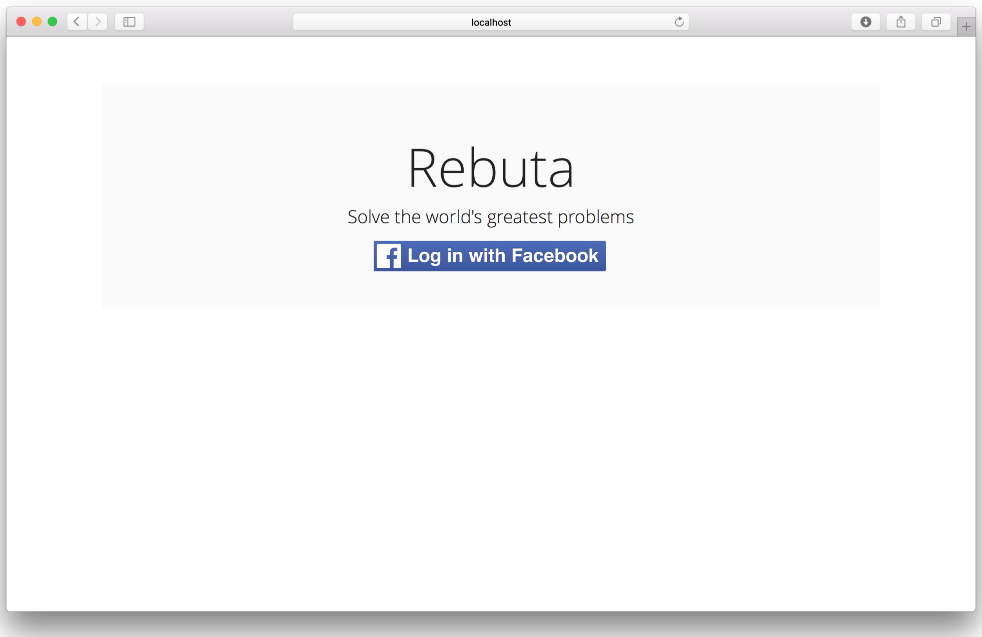
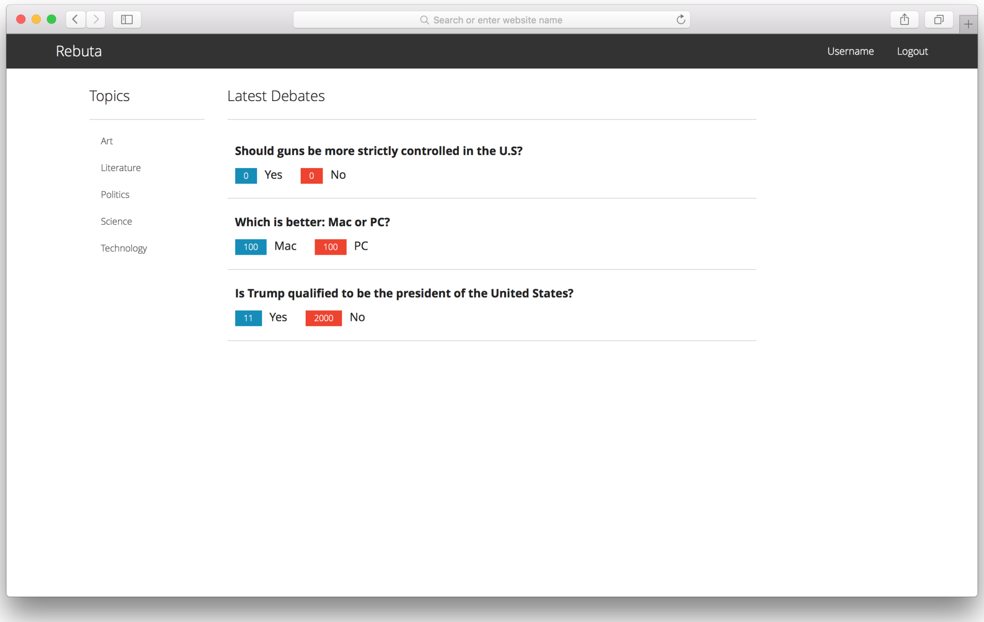


Figure 4: Landing page prompts users to log in with Facebook. This page is displayed if a user tries to access any page of the Rebuta site without being logged in.



Topics

- Art
- Literature
- Politics
- Science
- Technology

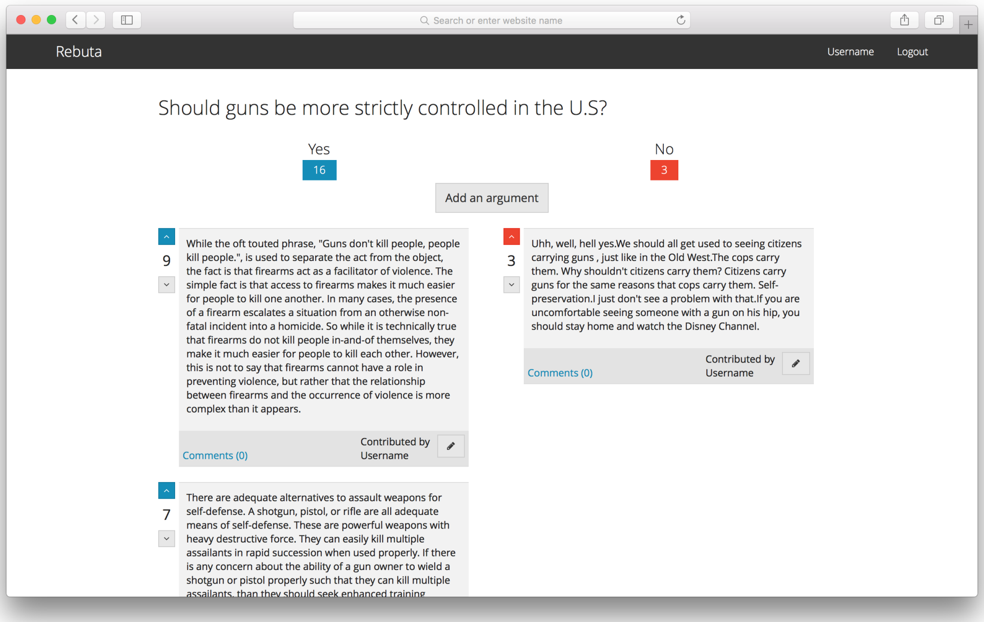
Latest Debates

**Should guns be more strictly controlled in the U.S?**
0 Yes 0 No

**Which is better: Mac or PC?**
100 Mac 100 PC

**Is Trump qualified to be the president of the United States?**
11 Yes 2000 No

Figure 5: Once logged in, the home page is the dashboard, which features a feed that displays all debates. Clicking on a debate takes the user to the debate page.



Should guns be more strictly controlled in the U.S?

Yes 16 | No 3

9 While the oft touted phrase, "Guns don't kill people, people kill people.", is used to separate the act from the object, the fact is that firearms act as a facilitator of violence. The simple fact is that access to firearms makes it much easier for people to kill one another. In many cases, the presence of a firearm escalates a situation from an otherwise non-fatal incident into a homicide. So while it is technically true that firearms do not kill people in-and-of themselves, they make it much easier for people to kill each other. However, this is not to say that firearms cannot have a role in preventing violence, but rather that the relationship between firearms and the occurrence of violence is more complex than it appears.

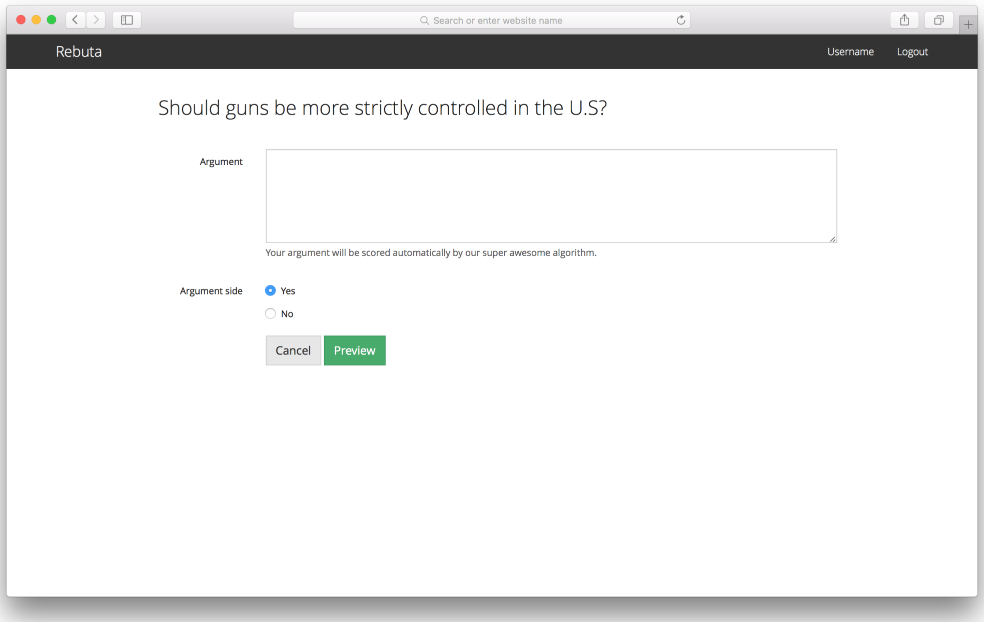
Comments (0) Contributed by Username

7 There are adequate alternatives to assault weapons for self-defense. A shotgun, pistol, or rifle are all adequate means of self-defense. These are powerful weapons with heavy destructive force. They can easily kill multiple assailants in rapid succession when used properly. If there is any concern about the ability of a gun owner to wield a shotgun or pistol properly such that they can kill multiple assailants, than they should seek enhanced training

3 Uhh, well, hell yes.We should all get used to seeing citizens carrying guns , just like in the Old West.The cops carry them. Why shouldn't citizens carry them? Citizens carry guns for the same reasons that cops carry them. Self-preservation.I just don't see a problem with that.If you are uncomfortable seeing someone with a gun on his hip, you should stay home and watch the Disney Channel.

Comments (0) Contributed by Username

Figure 6: The debate page has two columns that display debate arguments sorted by score. An initial score is assigned by the algorithm, and users can click to add upvotes and downvotes that affect the total score. To add an argument, the user can click the "Add an argument" button.



Should guns be more strictly controlled in the U.S?

Argument

Your argument will be scored automatically by our super awesome algorithm.

Argument side [✓ Yes] No

Figure 7: After clicking the "Add an argument" button, the user is brought to a page where s/he may type a response and choose a side on which the response will appear.