Journal of Artificial Intelligence Research 68 (2020) 607-632 Submitted 09/2019; published 07/2020

# A Multimodal Approach to Assessing Document Quality

**Aili Shen** AILIS@STUDENT.UNIMELB.EDU.AU
**Bahar Salehi** BAHAR.SALEHI@GMAIL.COM
**Jianzhong Qi** JIANZHONG.QI@UNIMELB.EDU.AU
**Timothy Baldwin** TB@LDWIN.NET
*School of Computing and Information Systems*
*The University of Melbourne*
*Victoria 3010, Australia*

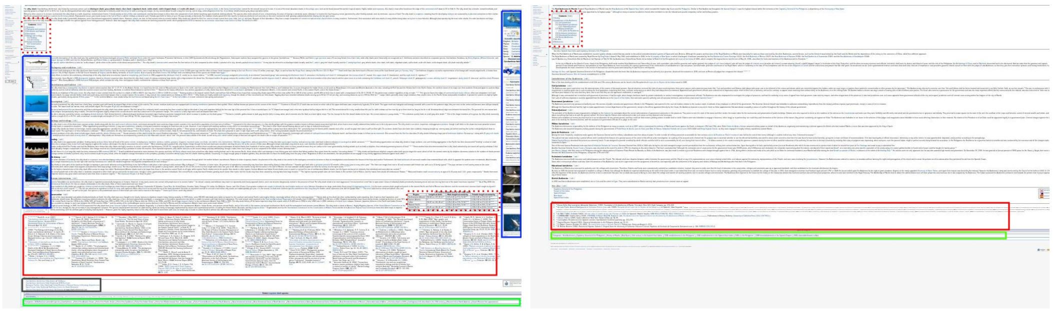
## Abstract

The perceived quality of a document is affected by various factors, including grammaticality, readability, stylistics, and expertise depth, making the task of document quality assessment a complex one. In this paper, we explore this task in the context of assessing the quality of Wikipedia articles and academic papers. Observing that the visual rendering of a document can capture implicit quality indicators that are not present in the document text — such as images, font choices, and visual layout — we propose a joint model that combines the text content with a visual rendering of the document for document quality assessment. Our joint model achieves state-of-the-art results over five datasets in two domains (Wikipedia and academic papers), which demonstrates the complementarity of textual and visual features, and the general applicability of our model. To examine what kinds of features our model has learned, we further train our model in a multi-task learning setting, where document quality assessment is the primary task and feature learning is an auxiliary task. Experimental results show that visual embeddings are better at learning structural features while textual embeddings are better at learning readability scores, which further verifies the complementarity of visual and textual features.

## 1. Introduction

The task of document quality assessment is to automatically assess a document according to some predefined inventory of quality labels. This can take many forms, including essay scoring (quality = language quality, coherence, and relevance to a topic), job application filtering (quality = suitability for role + visual/presentational quality of the application), or answer selection in community question answering (quality = actionability + relevance of the answer to the question). In this paper, we focus on document quality assessment in two contexts: Wikipedia document quality classification, and whether a paper submitted to a conference will be accepted or not.

Automatic quality assessment has obvious benefits in terms of time savings and tractability in contexts where the volume of documents is large. In the case of dynamic documents (possibly with multiple authors), such as in the case of Wikipedia, it is particularly pertinent, as any edit potentially has implications for the quality label of that document (and around 10 English Wikipedia documents are edited per second (Statistics, 2020)). Furthermore, when the quality assessment task is decentralized (as in the case of Wikipedia and academic paper assessment), quality criteria are often applied inconsistently by different



(a) Featured article

(b) Lower quality article

Figure 1: Visual renderings of two example Wikipedia documents with different quality labels (not intended to be readable).

people, where an automatic document quality assessment system could potentially reduce inconsistencies and enable immediate author feedback.

Current studies on document quality assessment mainly focus on textual features. For example, Warncke-Wang et al. (2015) examine features such as the article length and the number of headings to predict the quality class of a Wikipedia article. In contrast to these studies, in this paper, we propose to combine text features with visual features, based on a visual rendering of the document. Figure 1 illustrates our intuition, relative to Wikipedia articles. Without reading the text, we can tell that the article in Figure 1a is most likely to have a higher quality than that in Figure 1b, as it has a variety of images (in blue box), extensive references (in red box), and a list of categories (in green box). Based on this intuition, we aim to answer the following question: *can we achieve better accuracy at document quality assessment by complementing textual features with visual features?*

Our visual model is based on fine-tuning an Inception V3 model (Szegedy et al., 2016) over visual renderings of documents, while our textual model is based on a hierarchical bidirectional LSTM (Hochreiter & Schmidhuber, 1997). We further combine the two into a joint model. We perform experiments on five datasets: a Wikipedia dataset novel to this paper, three arXiv subsets split based on subject category (Kang et al., 2018), and a dataset consisting of academic papers in computer vision (Huang, 2018). Experimental results on the visual renderings of documents show that implicit quality indicators, such as images and visual layout, can be captured by an image classifier, at a level comparable to a text classifier. When we combine the two models, we achieve state-of-the-art results over four out of five of the datasets.

Modern-day neural network models are highly complex with millions or billions of parameters, hindering the explainability as to what kinds of features they have implicitly learned. This raises the following question: *what kinds of features does a textual model/visual model learn in predicting document quality?* To answer this question, we train our model to jointly predict document quality and a rich set of hand-crafted features, via multi-task learning. In doing so, we are able to analyse the behavior of neural networks in terms of their ability to reproduce features that have been shown to have utility in document quality assessment. Experimental results show that both the visual and textual model are generally more adept at learning structural features than readability scores. Analysing the results in detail, we find that the visual model is better at capturing structural features while the textual model is better at learning text readability scores.

This paper makes the following contributions:

(i) this is the first study to use visual renderings of documents to capture implicit quality indicators not present in the document text, such as document visual layout; experimental results show that we can obtain at least a 2.2% higher accuracy using only visual renderings of documents compared with using only textual features over a Wikipedia dataset, and we can obtain competitive results over an arXiv dataset.

(ii) we further propose a joint model to predict document quality combining visual and textual features; we observe further improvements on the Wikipedia dataset, two out of the three arXiv subsets, and the computer vision dataset, indicating complementarity between visual and textual features, and the general applicability of our proposed model.

(iii) this is the first study to explain what kinds of features a neural model for document quality assessment can learn; experimental results in a multi-task setting show that visual models are better at capturing structural features and textual models are better at learning readability scores, confirming the complementarity between textual and visual models.

(iv) we construct a large-scale Wikipedia dataset with full textual data, visual renderings, and quality class labels; we supplement the existing arXiv datasets with visual renderings of each document; we also supplement the dataset in computer vision with textual data.

All code and data are available at `https://github.com/AiliAili/MultiModal`.

In our earlier work (Shen et al., 2019), we presented experimental results over Wikipedia and arXiv datasets and analysed the performance of the visual and joint models in terms of gradient-based class activation maps (Selvaraju et al., 2017) and confusion matrices, respectively. In this paper, we make new contributions: (1) we explore the interpretability of our models by allowing them to learn document quality and hand-crafted features simultaneously, which we model as a multi-task learning problem; (2) we provide a detailed analysis of our textual, visual, and joint models in terms of precision, recall, and F1 score, and visualize article representations obtained by these three models to explain the different behaviours of the models; (3) we perform experiments over an additional dataset, based on academic papers in computer vision; and (4) we compare our models with an additional baseline with an attention mechanism (Yang et al., 2016), which achieves competitive performance in document classification.

## 2. Related Work

A variety of approaches have been proposed for document quality assessment across different domains: Wikipedia article quality assessment, academic paper rating, content quality assessment in community question answering (cQA), and essay scoring. Among these approaches, some use hand-crafted features while others use neural networks to learn representations from documents. For each domain, we first briefly describe feature-based approaches, and then review neural network-based approaches.

### 2.1 Wikipedia Article Quality Assessment

Quality assessment of Wikipedia articles is the task of assigning a quality class label to a given Wikipedia article, mirroring the quality assessment process that the Wikipedia community carries out manually. Many approaches have been proposed that use features from the article itself, meta-data features (e.g., the editors, and Wikipedia article revision history), or a combination of the two. Article-internal features capture information such as whether an article is properly organized, with supporting evidence, and with appropriate terminology. For example, Lipka and Stein (2010) use writing styles represented by binarized character trigram features to identify featured articles. Warncke-Wang et al. (2013) and Warncke-Wang et al. (2015) explore the number of headings, images, and references in the article. Dang and Ignat (2016a) further explore nine readability scores, such as the percentage of difficult words in the document, to measure the quality of the article. Meta-data features, which are indirect indicators of article quality, are usually extracted from revision history, and the interaction between editors and articles. For example, one heuristic that has been proposed is that higher-quality articles have more edits (Dalip et al., 2017, 2014). Wang and Iwaihara (2011) use the percentage of registered editors and the total number of editors of an article. Article–editor dependencies have also been explored. For example, Stein and Hess (2007) use the authority of editors to measure the quality of Wikipedia articles, where the authority of editors is determined by the quality of articles they edit.

Deep learning approaches to predicting Wikipedia article quality have also been proposed. For example, Dang and Ignat (2016b) use a version of doc2vec (Le & Mikolov, 2014) to represent articles, and feed the document embeddings into a four hidden layer neural network. Shen et al. (2017) first obtain sentence representations by averaging words within a sentence. Then, they apply a bidirectional LSTM (Hochreiter & Schmidhuber, 1997) to learn a document-level representation, which is combined with hand-crafted features as side information. Dang and Ignat (2017) exploit two stacked biLSTMs to learn document representations. Wang and Li (2020) compare the performance of different neural models (such as convolutional neural networks “CNN” and LSTMs), in differentiating high quality Wikipedia articles from low quality ones. They do not perform any analysis of the learned representations, making it difficult to understand what the models have learned.

### 2.2 Academic Paper Rating

Academic paper rating is a relatively new task in NLP/AI, with the basic formulation being to automatically predict whether a paper is accepted/rejected by a conference, based on the naive assumption that any paper published at a workshop was/would have been rejected by a conference. Kang et al. (2018) explore hand-crafted features, such as the length of the title, whether specific words (such as *outperform*, *state-of-the-art*, and *novel*) appear in the abstract, and an embedded representation of the abstract as input to different downstream learners, such as logistic regression, decision tree, and random forest. Yang et al. (2018)

exploit a modularized hierarchical CNN, where each paper section is treated as a module. For each paper section, they train an attention-based CNN, and an attentive pooling layer is applied to the concatenated representation of each section, which is then fed into a softmax layer. Huang (2018) uses ResNet-18 (He et al., 2016) to learn visual features only (similar to our Inception baseline, which we introduce later in this paper, in the Wikipedia section), to predict whether a paper in computer vision area is a conference or workshop paper.

### 2.3 Content Quality Assessment in cQA

Automatic quality assessment in cQA is the task of determining whether an answer is of high quality, selected as the best answer, or ranked higher than other answers (Hoogeveen et al., 2018). To measure answer content quality in cQA, researchers have exploited various features from different sources, such as the answer content itself, the answerer's profile, interactions among users, and usage of the content. The most common feature used is the answer length (Jeon et al., 2006; Suryanto et al., 2009), with other features including: syntactic and semantic features, such as readability scores (Agichtein et al., 2008); similarity between the question and the answer at lexical, syntactic, and semantic levels (Agichtein et al., 2008; Belinkov et al., 2015; Hou et al., 2015); or user data (e.g., a user's reputation points or the number of answers written by the user: Hoogeveen et al. (2018)).

There have also been approaches using neural models to learn representations. For example, Suggu et al. (2016) combine CNN-learned representations with hand-crafted features to predict answer quality. Zhou et al. (2015) use a 2-dimensional CNN to learn the semantic relevance of an answer to the question, and apply an LSTM to the answer sequence to model thread context. Guzmán et al. (2016a) and Guzmán et al. (2016b) model the problem similarly to machine translation quality estimation, treating answers as competing translation hypotheses and the question as the reference translation, and apply a neural machine translation evaluation model to the problem.

### 2.4 Essay Scoring

Automated essay scoring is the task of assigning a score to an essay, usually in the context of assessing the language ability of a language learner. The quality of an essay is affected by the following four primary dimensions: topic relevance, organization and coherence, word usage and sentence complexity, and grammar and mechanics. To measure whether an essay is relevant to its "prompt" (the description of the essay topic), lexical and semantic overlap is commonly used (Persing & Ng, 2014; Phandi et al., 2015). Attali and Burstein (2004) explore word features, such as the number of verb formation errors, average word frequency, and average word length, to measure word usage and lexical complexity. Cummins et al. (2016) use sentence structure features to measure sentence variety. The effects of grammatical and mechanistic errors on the quality of an essay are measured via word and part-of-speech $n$-gram features and "mechanics" features (Persing & Ng, 2013) (e.g., spelling, capitalization, and punctuation), respectively. Taghipour and Ng (2016), Alikaniotis et al. (2016), and Tay et al. (2018) use an LSTM to obtain an essay representation, which is used as the basis for classification. Similarly, Dong et al. (2017) utilize a CNN to obtain sentence representation and an LSTM to obtain essay representation, with an attention layer at both the sentence and essay levels.

### 2.5 Multimodal Document Processing

There have also been studies exploring what kinds of features a model can learn when complementing textual information with visual information. For example, Bruni et al. (2014) complement the distributional representations of words with distributional representations of visual words extracted from a multimodal document containing both text and images. Their proposed multimodal method consists of two sub-models: a text model and a visual model. Each sub-model obtains semantic representations derived from treating words/visual words as a bag-of-words, ignoring order. In comparison, our work explores not only what kinds of features each sub-model can learn but also their correlation with hand-crafted features. Further, our textual sub-model takes word order into account.

In this paper we focus exclusively on the content of the document itself. Metadata related to the document, such as the revision history and editor–article network in Wikipedia, or co-authorship network in academic paper rating, are left for future exploration.

## 3. The Proposed Joint Model

We treat document quality assessment as a classification problem, i.e., given a document, we predict its quality class (e.g., whether an academic paper should be accepted or rejected). The proposed model is a joint model that integrates visual features learned through a visual model, with textual features learned through a textual model. In this section, we present the details of the visual and textual embeddings, and describe how we combine the two. We return to discussing hyper-parameter settings and the experimental configuration in the next section.

### 3.1 Visual Embedding Learning

A wide range of models have been proposed to tackle the image classification task, such as VGG (Simonyan & Zisserman, 2015), ResNet (He et al., 2016), Inception V3 (Szegedy et al., 2016), and Xception (Chollet, 2017). To the best of our knowledge, there is just one existing work that has considered visual renderings of documents for quality assessment, which is Huang (2018); it uses visual features only (similar to our Inception baseline in the Wikipedia section) to predict whether a paper is a conference or workshop paper. In our work, we use Inception V3 pretrained on ImageNet (ImageNet, 2020) ("Inception" hereafter) to obtain visual embeddings of documents, noting that any image classifier could be applied to our task. The input to Inception is a visual rendering (screenshot) of a document, and the output is a visual embedding, which we will later integrate with our textual embedding.

Based on the observation that it is difficult to decide what types of convolution to apply to each layer (such as 3×3 or 5×5), the basic Inception model applies multiple convolution filters in parallel and concatenates the resulting features, which are fed into the next layer. This has the benefit of capturing both local features through smaller convolutions and abstracted features through larger convolutions. Inception is a hybrid of multiple Inception models of different architectures. To reduce computational cost, Inception also modifies the basic model by applying a 1×1 convolution to the input and factorizing larger convolutions into smaller ones.

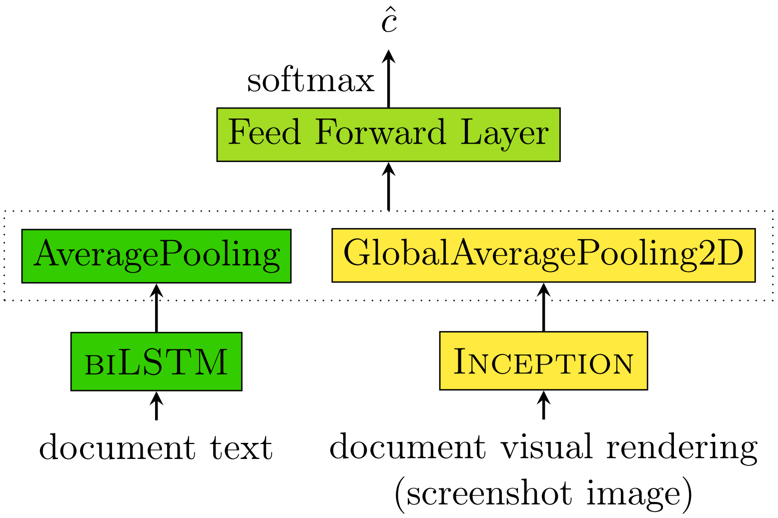


Figure 2: Overview of the proposed joint model.

### 3.2 Textual Embedding Learning

We adopt a bi-directional LSTM model to generate textual embeddings for document quality assessment, following the method of Shen et al. (2017) ("biLSTM" hereafter). The input to biLSTM is a textual document, and the output is a textual embedding, which we will later integrate with the visual embedding.

For biLSTM, each word is represented as a word embedding (Bengio et al., 2003), and an average-pooling layer is applied to the word embeddings to obtain the sentence embedding, which is fed into a bi-directional LSTM. Then an average-pooling layer is applied to obtain document representations.

### 3.3 The Joint Model

The proposed joint model ("Joint" hereafter) combines the visual and textual embeddings (output of Inception and biLSTM) via a simple feed-forward layer and softmax over the document label set, as shown in Figure 2. We optimize our model based on cross-entropy loss.

## 4. Experimental Studies

In this section, we first describe our experimental settings. Then, we report experimental results over five datasets: (1) Wikipedia, (2) three arXiv subsets, and (3) CVPG. In each case, we first describe the dataset, then present results, including multiple baseline approaches in each case. Note that all five datasets are based in English, although we expect our models to be readily applicable to different languages, as the only language-specific components are the word tokenizer and pre-trained word embeddings.

### 4.1 Experimental Setting

As discussed above, our model has two main components — biLSTM[1] and Inception— which generate textual and visual representations, respectively. For the biLSTM component, the documents are preprocessed as described in Shen et al. (2017), where an article is divided into sentences and tokenized using NLTK (Bird, 2006). Words appearing more than 20 times are retained when building the vocabulary. All other words are replaced by the special UNK token. We use the pre-trained GloVe (Pennington et al., 2014) 50-dimensional word embeddings to represent words. For words not in GloVe, word embeddings are randomly initialized based on sampling from a uniform distribution $U(-1, 1)$. All word embeddings are updated in the training process. We set the LSTM hidden layer size to 256. The concatenation of the forward and backward LSTMs thus gives us 512 dimensions for the document embedding. A dropout layer is applied at both the sentence and document level, respectively, with a probability of 0.5.

For Inception, we adopt data augmentation techniques in the training with a "nearest" filling mode, a zoom range of 0.1, a width shift range of 0.1, and a height shift range of 0.1. As the original screenshots are of size 1,000×2,000 pixels, they are resized to 500×500 to feed into Inception, where the input shape is (500, 500, 3). A dropout layer is applied with a probability of 0.5. Then, a GlobalAveragePooling2D layer is applied, which produces a 2,048 dimensional representation.

For the Joint model, we get a representation of 2,560 dimensions by concatenating the 512 dimensional representation from biLSTM with the 2,048 dimensional representation from Inception. The dropout layer is applied to the two components with a probability of 0.5. For biLSTM, we use a mini-batch size of 128 and a learning rate of 0.001. For both Inception and the Joint model, we use a mini-batch size of 16 and a learning rate of 0.0001. All hyper-parameters were set empirically over the development data, and the models are optimized using the Adam optimizer (Kingma & Ba, 2015).

In the training phase, the weights in Inception are initialized by parameters pretrained on ImageNet, and the weights in biLSTM are randomly initialized (except for the word embeddings). We train each model for 50 epochs. However, to prevent overfitting, we adopt early stopping, where we stop training the model if the performance on the development set does not improve for 20 epochs. For evaluation, we use (micro-)accuracy, following previous studies (Dang & Ignat, 2016a; Kang et al., 2018; Huang, 2018).

### 4.2 Wikipedia

In this section, we first describe the Wikipedia dataset used to benchmark our method, followed by baselines and experimental results.

#### 4.2.1 Dataset

The Wikipedia dataset consists of articles from English Wikipedia, with quality class labels assigned by the Wikipedia community. Wikipedia articles are labelled with one of six quality classes, in descending order of quality: Featured Article ("FA"), Good Article ("GA"), B-class

[1] We adopt biLSTM rather than biLSTM$_\text{H}$ (introduced later) in our Joint model as biLSTM achieves better performance than biLSTM$_\text{H}$ in general, which can be observed in our experimental results.

| Class | Train | Dev | Test | Total |
|---|---|---|---|---|
| FA | 4000 | 500 | 500 | 5000 |
| GA | 4000 | 500 | 500 | 5000 |
| B | 4000 | 500 | 455 | 4955 |
| C | 4000 | 500 | 467 | 4967 |
| Start | 4000 | 500 | 451 | 4951 |
| Stub | 4000 | 500 | 421 | 4921 |
| Total | 24000 | 3000 | 2794 | 29794 |

Table 1: The composition of the Wikipedia dataset, in terms of numbers of documents of the different quality classes.

Article ("B"), C-class Article ("C"), Start Article ("Start"), and Stub Article ("Stub"). A description of the criteria associated with the different classes can be found in the Wikipedia grading scheme page (Scheme, 2020). The quality class of a Wikipedia article is assigned by Wikipedia reviewers or any registered user, who can discuss through the article's talk page (Talk, 2020) to reach a consensus. We constructed the dataset by first crawling all articles from each quality class repository, e.g., we get FA articles by crawling articles from the FA repository: `https://en.wikipedia.org/wiki/Category:Featured_articles`. This resulted in around 5K FA, 28K GA, 212K B, 533K C, 2.6M Start, and 3.2M Stub articles.

We randomly sampled 5,000 articles from each quality class and removed all redirect pages, resulting in a dataset of 29,794 articles. As the wikitext contained in each document contains markup relating to the document category such as *{Featured Article}* or *{geo-stub}*, which reveals the quality label, we removed such information. We additionally randomly partitioned this dataset into training, development, and test splits based on a ratio of 8:1:1. Details of the dataset are presented in Table 1.

We generate a visual representation of each document via a 1,000×2,000-pixel screenshot of the article by running a PhantomJS script (Rendering, 2020) over the rendered version of the article, ensuring that the screenshot and wikitext versions of the article are the same version. Any direct indicators of document quality (such as the FA indicator, which is a bronze star icon in the top right corner of the web page) are removed from the screenshot.

### 4.2.2 Baseline Approaches

We compare our models against the following six baselines:

- Majority: the model labels all test samples with the majority class of the training data.

- Benchmark: a benchmark method from the literature (Dang & Ignat, 2016a), which uses structural features and readability scores as features to build a random forest classifier. Detailed description of these features can be found in the Hand-Crafted Features section.

| | | Majority | Benchmark | Doc2Vec | Inception$_{\text{fixed}}$ | biLSTM$_{\text{H}}$ | biLSTM | Inception | Joint |
|---|---|---|---|---|---|---|---|---|---|
| Wikipedia | | 16.7% | 46.7±0.34% | 23.2±1.41% | 43.7±0.51% | 54.8±0.68% | 54.1±0.47% | 57.0±0.63% | **59.4±0.47%**$^{\dagger}$ |
| arXiv | cs.ai | 92.2% | 92.6% | 73.3±9.81% | 92.3±0.29% | 92.2±1.28% | 91.5±1.03% | 92.8±0.79% | **93.4±1.07%**$^{\dagger}$ |
| | cs.cl | 68.9% | 75.7% | 66.2±8.38% | 75.0±1.95% | 73.3±3.41% | 76.2±1.30% | 76.2±2.92% | **77.1±3.10%** |
| | cs.lg | 67.9% | 70.7% | 64.7±9.08% | 73.9±1.23% | 76.8±2.11% | **81.1±0.83%** | 79.3±2.94% | 79.9±2.54% |
| CVPG | | 79.33% | N/A | 72.3±10.04% | 82.9±0.40% | 87.7±0.39% | 88.2±0.49% | 86.9±0.87% | **88.8±0.90%** |

Table 2: Experimental results over the 5 datasets, averaged across 10 runs. The best result for each dataset is indicated in **bold**, and marked with "†" if it is significantly better than the second best result (based on a one-tailed Wilcoxon signed-rank test; $p < 0.05$). The results of Benchmark on the arXiv dataset are from the original paper, where the standard deviation values were not reported. All neural models except for Inception$_{\text{fixed}}$ have larger standard deviation values on arXiv than Wikipedia and CVPG, which can be explained by the small size of the arXiv test set.

- Doc2Vec: doc2vec (Le & Mikolov, 2014; Lau & Baldwin, 2016) to learn 500d document embeddings, and a 4-layer feed-forward classification model on top of this, with 2000, 1000, 500, and 200 dimensions, respectively.
- Inception$_{\text{fixed}}$: the frozen Inception model, where only parameters in the last layer are fine-tuned during training.
- biLSTM$_{\text{H}}$: the hierarchical attention model of Yang et al. (2016), which incorporates hierarchical structure and an attention mechanism into a GRU (Cho et al., 2014), and achieves improved performance over six document classification datasets, such as Yelp (Tang et al., 2015) and Amazon (Zhang et al., 2015). Thus we also compare our proposed models with a biLSTM, with a hierarchical structure and attention mechanism ("biLSTM$_{\text{H}}$" hereafter). biLSTM$_{\text{H}}$ first generates a sentence embedding by applying a biLSTM to words in a sentence, then an attention mechanism to outputs of the biLSTM to weight words based on their importance in the sentence; then the sequence of sentences is fed into another biLSTM, where another attention layer is used to weight outputs based on the importance of sentences in the document. Here, document tokenization, word embedding initialization and updating are the same as in biLSTM. We set the biLSTM hidden layer size for both the sentence and document levels to 50, which gives us a 100-dimensional sentence and document embedding. The context vectors for both words and sentences also have a dimensionality of 100. A dropout layer is applied at both the sentence and document levels, with a probability of 0.5. All other hyper-parameters are set as in biLSTM except that a mini-batch size of 64 is used in biLSTM$_{\text{H}}$.
- biLSTM: first derive a sentence representation by averaging across words in a sentence, then feed the sentence representation into a biLSTM and an average-pooling layer over the output sequence to learn a 512d document level representation, which is used to predict document quality.

The hyper-parameters of Benchmark, Doc2Vec, and biLSTM are based on the corresponding papers, except that we fine-tune the feed forward layer of Doc2Vec on the development set and train the model for 300 epochs.

| Subject | Accepted | Train | Dev | Test | Total |
|---|---|---|---|---|---|
| cs.ai | 10% | 3682 | 205 | 205 | 4092 |
| cs.cl | 30% | 2374 | 132 | 132 | 2638 |
| cs.lg | 32% | 4543 | 252 | 253 | 5048 |

Table 3: The composition of the arXiv dataset. "Accepted" indicates the proportion of accepted papers in the given subject.

#### 4.2.3 Experimental Results

Table 2 shows the performance of the different models, in the form of the average accuracy on the test set (along with the standard deviation) over 10 runs, with different random initializations.

On Wikipedia, we observe that the performance of biLSTM$_{\mathrm{H}}$, biLSTM, Inception, and Joint is much better than that of all four baselines. Inception achieves 2.9% higher accuracy than biLSTM and 2.2% higher accuracy than biLSTM$_{\mathrm{H}}$. The performance of Joint achieves an accuracy of 59.4%, which is at least 4.6% higher than using textual features alone (biLSTM and biLSTM$_{\mathrm{H}}$) and 2.4% higher than using visual features alone (Inception). Based on a one-tailed Wilcoxon signed-rank test, the performance of Joint is statistically significant ($p < 0.05$). This shows that the textual and visual features complement each other, achieving state-of-the-art results in combination.

### 4.3 Arxiv

In this section, we describe the arXiv dataset, followed by baselines and experimental results.

#### 4.3.1 Dataset

The arXiv dataset (Kang et al., 2018) consists of three subsets of academic papers under the arXiv repository of Computer Science (cs), from the three subject areas of: Artificial Intelligence (cs.ai), Computation and Language (cs.cl), and Machine Learning (cs.lg). In line with the original dataset formulation (Kang et al., 2018), a paper is considered to have been accepted (i.e. is positively labeled) if it matches a paper in the DBLP database or is otherwise accepted by any of the following conferences: ACL, EMNLP, NAACL, EACL, TACL, NeurIPS, ICML, ICLR, or AAAI. Failing this, it is considered to be rejected (noting that, in practice, some of the papers may not have been submitted to any of these conferences). The median numbers of pages for papers in cs.ai, cs.cl, and cs.lg are 11, 10, and 12, respectively. To make sure each page in the PDF file has the same size in the screenshot, we crop the PDF file of a paper to the first 12; we pad the PDF file with blank pages if a PDF file has less than 12 pages, using the PyPDF2 Python package (PyPDF2, 2020). We then use ImageMagick (ImageMagick, 2020) to convert the 12-page PDF file to a single 1,000×2,000 pixel screenshot. Table 3 details this dataset, where the "Accepted" column denotes the percentage of positive instances (accepted papers) in each subset.

### 4.3.2 Baseline Approaches

We compare our models against the six baselines mentioned above except that: (1) Benchmark is the method of Kang et al. (2018), who use hand-crafted features, such as the number of references and TF-IDF weighted bag-of-words in the abstract, to build a classifier based on the best of logistic regression, multi-layer perception, and AdaBoost; (2) we fine-tune the feed forward layer of Doc2Vec on the development set and train the model for 50 epochs.

### 4.3.3 Experimental Results

Once again, the Joint model achieves the highest accuracy on cs.ai and cs.cl by combining textual and visual representations (at a level of statistical significance for cs.ai). This, again, confirms that textual and visual features complement each other, and together they achieve state-of-the-art results. On cs.lg, Joint achieves a 0.6% higher accuracy than Inception by combining visual features and textual features, but biLSTM achieves the highest accuracy. One characteristic of cs.lg documents is that they tend to contain more equations than the other two arXiv datasets, and preliminary analysis suggests that the biLSTM is picking up on a correlation between the volume/style of mathematical presentation and the quality of the document. Surprisingly, $\text{Inception}_{\text{fixed}}$ is better than Majority and Benchmark over the arXiv cs.lg subset, which verifies the usefulness of visual features, even when only the last layer is fine-tuned. Table 2 also shows that Inception and biLSTM achieve similar performance on arXiv, showing that textual and visual representations are equally discriminative: Inception and biLSTM are indistinguishable over cs.cl; biLSTM achieves 1.8% higher accuracy over cs.lg, while Inception achieves 1.3% higher accuracy over cs.ai. Comparisons between $\text{biLSTM}_{\text{H}}$ and Inception show the superiority of Inception: Inception achieves 0.6%, 2.9%, and 2.5% higher accuracy over cs.ai, cs.cl, and cs.lg, respectively.

For cs.ai, Majority, Benchmark, and $\text{Inception}_{\text{fixed}}$ outperform or are competitive with biLSTM and $\text{biLSTM}_{\text{H}}$, in large part because of the class imbalance in this dataset (90% of papers are rejected).

## 4.4 CVPG

In this section, we describe the CVPG dataset, followed by baselines and experimental results.

### 4.4.1 Dataset

The CVPG dataset (Huang, 2018) consists of conference papers ("accepted" papers) and workshop papers ("rejected" papers) from top-tier computer vision conferences: six CVPR and three ICCV proceedings from 2013 to 2018. In line with Huang (2018), a paper is considered to be accepted if it is from the main conference, and otherwise is considered to have been rejected (from the main conference). Following Huang (2018), we crop/pad the PDF file of a paper to 8 pages if a paper has more/less than 8 pages.[2] We then use ImageMagick to convert the 8-page pdf file to a single 1,000×2,000 pixel screenshot. Main conference papers and workshop papers from CVPR 2017 are used as the validation set; main

[2] Workshop papers with less than 7 pages are removed, as they make the classification task trivial for the visual model.

| Class | Train | Dev | Test | Total |
|---|---|---|---|---|
| Accepted | 3856 | 783 | 978 | 5617 |
| Rejected | 1005 | 251 | 247 | 1503 |
| Total | 4861 | 1034 | 1225 | 7120 |

Table 4: The composition of the CVPG dataset, in terms of the number of documents that were accepted or rejected by a conference.

conference papers and workshop papers from CVPR 2018 are the test set; the remaining papers are the training set. Full statistics of the dataset are shown in Table 4.

#### 4.4.2 Baseline Approaches

Over CVPG, we compare our models against the same baselines used in Wikipedia. The Benchmark baseline is not provided for CVPG, as it was not possible for us to extract hand-crafted features from the parsed PDF files with high precision: we first extracted the textual content from each paper using the Science Parse library (Parse, 2020), then extracted hand-crafted features from the parsed content. However, the Science Parse library makes mistakes, resulting in missing sections — such as the abstract — in the extracted content, making it difficult for us to extract hand-crafted features.[3] The hyper-parameters of the baselines are the same as for Wikipedia except that we train Doc2Vec for 50 epochs.

#### 4.4.3 Experimental Results

In the case of the CVPG dataset, biLSTM$_{\mathrm{H}}$, biLSTM, Inception, and Joint achieve substantial improvements over Majority and Doc2Vec. The usefulness of visual features learned by Inception$_{\text{FIXED}}$ (fine-tuning only the last layer of the network) is once again verified over this dataset: Inception$_{\text{FIXED}}$ obtains superior performance over Majority and Doc2Vec. The Joint model achieves the best performance by combining textual features and visual features, which demonstrates that textual and visual features complement each other. Similar to cs.lg, biLSTM outperforms Inception over the CVPG dataset. Preliminary analysis over the CVPG dataset indicates that mathematical presentations are quite common, although less prevalent on average than in cs.lg and with greater variance in their prevalence. As such biLSTM achieves better performance by picking up on the correlation between mathematical presentations and the quality of a document. Furthermore, images in CVPG documents are quite prevalent, contributing to the superior performance of Joint by combining the textual and visual representations.

### 4.5 Summary of Results across Datasets

The model performance across the different datasets is not directly comparable, as these datasets are intrinsically different: (1) Wikipedia is a six-class classification task, while arXiv and CVPG are binary; and (2) the proportion of positive (accepted) instances in cs.ai, cs.cl,

[3] For arXiv, it contains additional review files including the abstract, making it possible for us to extract hand-crafted features mainly from the abstract.

cs.lg, and CVPG differ considerably (10%, 30%, 32%, and 80%, respectively). Having said this, we can compare the performance relative to the respective baselines and within task. As shown in Table 2, Inception and biLSTM are competitive with each other across all these datasets: Inception and biLSTM achieve the same performance over cs.cl; biLSTM is superior to Inception over cs.lg and CVPG, while Inception outperforms biLSTM over Wikipedia and cs.ai. The Joint model achieves state-of-the-art performance over all the datasets except for cs.lg. This affirms the complementarity of visual and textual features. The superior performance of the Joint model over the different datasets also attests to the general applicability of our model.

## 5. Analysis

In this section, we first analyze the performance of Inception and Joint. We also analyze the performance of different models on different quality classes. Additionally, we compare the high-level representations learned by the different models through visualization. As the Wikipedia test set is larger and more balanced than those of arXiv and CVPG, our analysis will focus on Wikipedia.

### 5.1 Inception

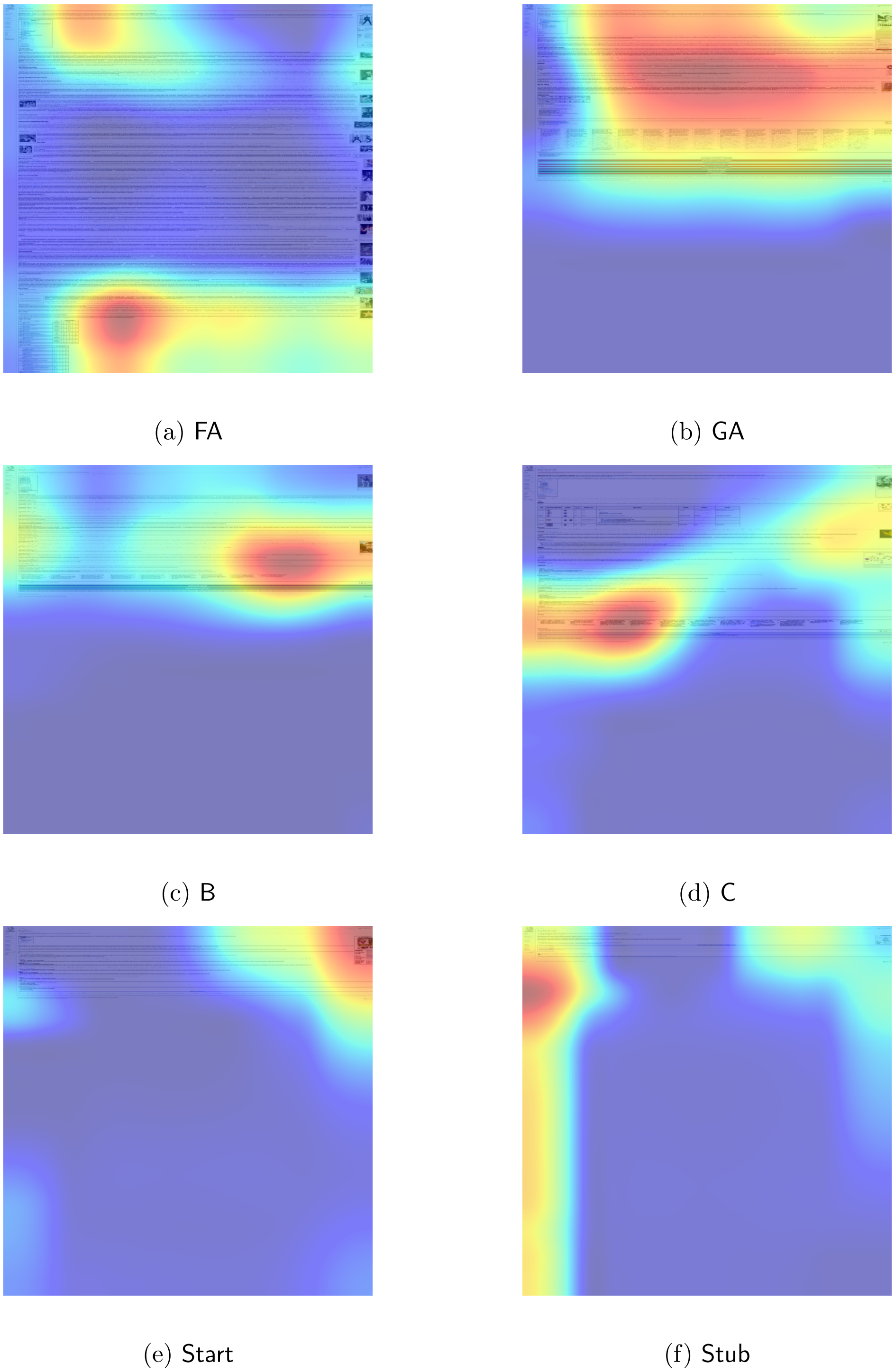
To better understand the performance of Inception, we generated the gradient-based class activation map (Selvaraju et al., 2017), by maximizing the outputs of each class in the penultimate layer, as shown in Figure 3. From Figure 3a and Figure 3b, we can see that Inception identifies the two most important regions (one at the top corresponding to the table of contents, and the other at the bottom, capturing both document length and tables) that contribute to the FA class prediction, and a region in the upper half of the image that contributes to the GA class prediction (capturing images and the length of the article body). From Figure 3c and Figure 3d, we can see that the most important regions in terms of B and C class prediction capture images (down the left and right of the page, in the case of B and C), and document length and references. From Figure 3e and Figure 3f, we can see that Inception finds that images in the top right corner are the strongest predictor of Start class prediction, and (the lack of) images/the link bar down the left side of the document are the most important for Stub class prediction.

### 5.2 Joint

Table 5 shows the confusion matrix of Joint on Wikipedia. We can see that more than 50% of documents for each quality class are correctly classified, except for the C class where more documents are misclassified into B. Analysis shows that when misclassified, documents are usually misclassified into adjacent quality classes, which can be explained by the Wikipedia grading scheme, where the criteria for adjacent quality classes are more similar.[4]

We also provide a breakdown of precision ("$\mathcal{P}$"), recall ("$\mathcal{R}$"), and F1 score ("$\mathcal{F}_{\beta=1}$") for biLSTM, Inception, and Joint across the quality classes in Table 6. We can see that Joint achieves the highest accuracy in 11 out of 18 cases. It is also worth noting that all

[4] Suggesting that ordinal regression should boost accuracy, but preliminary experiments with various methods led to no improvement over simple classification.



(a) FA

(b) GA

(c) B

(d) C

(e) Start

(f) Stub

Figure 3: Heatmap overlapped onto screenshots of each Wikipedia quality class. Best viewed in color.

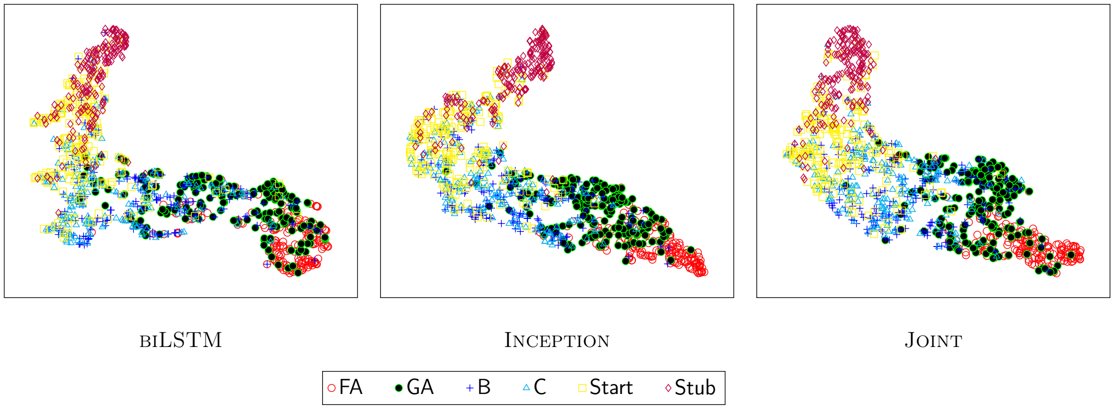
| Quality | FA | GA | B | C | Start | Stub |
|---|---|---|---|---|---|---|
| FA | 397 | 83 | 20 | 0 | 0 | 0 |
| GA | 112 | 299 | 65 | 22 | 2 | 0 |
| B | 23 | 53 | 253 | 75 | 44 | 7 |
| C | 5 | 33 | 193 | 124 | 100 | 12 |
| Start | 1 | 6 | 36 | 85 | 239 | 84 |
| Stub | 0 | 0 | 6 | 7 | 63 | 345 |

Table 5: Confusion matrix of the Joint model on Wikipedia. Rows are the actual quality classes and columns are the predicted quality classes. The diagonal (gray cells) indicates correct predictions.

| Quality | Metric | biLSTM | Inception | Joint |
|---|---|---|---|---|
| FA | $\mathcal{P}$ | **76.6** | 74.8 | 73.8 |
|  | $\mathcal{R}$ | 72.0 | 68.2 | **79.4** |
|  | $\mathcal{F}_{\beta=1}$ | 74.2 | 71.3 | **76.5** |
| GA | $\mathcal{P}$ | 51.3 | 57.7 | **63.1** |
|  | $\mathcal{R}$ | **59.8** | 59.0 | **59.8** |
|  | $\mathcal{F}_{\beta=1}$ | 55.2 | 58.3 | **61.4** |
| B | $\mathcal{P}$ | 37.6 | 41.8 | **44.2** |
|  | $\mathcal{R}$ | 42.4 | 44.0 | **55.6** |
|  | $\mathcal{F}_{\beta=1}$ | 39.9 | 42.9 | **49.2** |
| C | $\mathcal{P}$ | 36.3 | 38.9 | **39.6** |
|  | $\mathcal{R}$ | 27.0 | **36.0** | 26.6 |
|  | $\mathcal{F}_{\beta=1}$ | 31.0 | **37.4** | 31.8 |
| Start | $\mathcal{P}$ | 48.2 | 49.4 | **53.3** |
|  | $\mathcal{R}$ | 44.8 | **57.2** | 53.0 |
|  | $\mathcal{F}_{\beta=1}$ | 46.4 | 53.0 | **53.1** |
| Stub | $\mathcal{P}$ | 71.9 | **83.3** | 77.0 |
|  | $\mathcal{R}$ | 78.9 | 78.2 | **81.9** |
|  | $\mathcal{F}_{\beta=1}$ | 75.2 | **80.7** | 79.4 |

Table 6: Precision ("$\mathcal{P}$"), recall ("$\mathcal{R}$"), and F1 ("$\mathcal{F}_{\beta=1}$") of biLSTM, Inception, and Joint on Wikipedia.

models achieve higher scores for FA, GA, and Stub articles than B, C and Start articles. This can be explained in part by the fact that FA and GA articles must pass an official review based on structured criteria, and in part by the fact that Stub articles are usually very short, which is discriminative for Inception, and Joint. All models perform worst on the B and C quality classes. It is difficult to differentiate B articles from C articles even for Wikipedia contributors. As evidence of this, when we crawled a new dataset including talk pages with quality class votes from Wikipedia contributors, we found that among articles with three



biLSTM

Inception

Joint

Figure 4: t-SNE scatter plot of Wikipedia article representations (representations from the penultimate layer of each model, based on 200 random samples from each quality class; best viewed in color)

or more quality labels, over 20% percent of B and C articles have inconsistent votes from Wikipedia contributors, whereas for FA and GA articles the number is only 0.7%.

We further visualize the learned document representations of biLSTM, Inception, and Joint in the form of a t-SNE plot (van der Maaten & Hinton, 2008) in Figure 4. The degree of separation between Start and Stub achieved by Inception is much greater than for biLSTM, with the separation between Start and Stub achieved by Joint being the clearest among the three models. Inception and Joint are better than biLSTM at separating Start and C. Joint achieves slightly better performance than Inception in separating GA and FA. We can also see that it is difficult for all models to separate B and C, which is consistent with the findings of Tables 5 and 6.

## 6. Model Interpretability

Although neural networks have achieved competitive or state-of-the-art performance across various tasks, such as document quality assessment and essay scoring, they have been criticized for lacking explainability. In this section, we explore what kinds of features biLSTM and Inception implicitly learn, by allowing them to learn the quality of documents and hand-crafted feature values simultaneously. In other words, we model this as a multi-task learning problem: (1) one task is to learn the document quality; and (2) the second task is to learn the value of a given feature. We first detail 21 features used in the literature (Dang & Ignat, 2016a) to assess the quality of Wikipedia articles, then present experimental results of multi-task learning over the Wikipedia dataset. We also study the performance difference of biLSTM, Inception, and Joint by combining neural network-learned features with hand-crafted features.

| Structural Features | Readability Scores |
|---|---|
| Article length in bytes (*Length*) | Flesch reading score (Kincaid et al., 1975) (*Flesch*) |
| Number of references (*References*) | Flesch-Kincaid grade level (Kincaid et al., 1975) (*Flesch-Kincaid*) |
| Number of links to other Wikipedia pages (*Pagelinks*) | Smog index (McLaughlin, 1969) (*Smog*) |
| Number of citation templates (*Citation*) | Coleman-Liau index (Coleman & Liau, 1975) (*Coleman-Liau*) |
| Number of non-citation templates (*Non-citation*) | Automated readability index (Smith & Senter, 1967) (*Readability Index*) |
| Number of categories linked in the text (*Categories*) | Difficult words (Chall & Dale, 1995) (*Difficult Words*) |
| Number of images / length of the article (*Images*) | Dale-Chall score (Dale & Chall, 1948) (*Dale-Chall*) |
| Article having an infobox or not (*Infobox*) | Linsear write formula (Chen, 2012) (*Linsear*) |
| Number of level 2 headings (*Level2*) | Gunning-Fog index (Gunning, 1969) (*Gunning-Fog*) |
| Number of level 3+ headings (*Level3+*) | Information noise score (Zhu & Gauch, 2000) (*Infonoise*) |
|  | Readability consensus (*Consensus*) |

Table 7: Hand-crafted features. Here, the text in parentheses (e.g., "*Length*") is a descriptor for each feature, which will be used hereafter.

### 6.1 Hand-Crafted Features

Following Dang and Ignat (2016a), we use structural features and readability scores as hand-crafted features for quality class prediction. The structural features capture the structure of articles, and the readability scores reflect writing style. These are extracted from Wikipedia articles using the open-source packages wikiclass (WikiClass, 2020) and textstat (TextStat, 2020). The features are listed in Table 7.

The structural features reflect article quality in different ways. For example, *article length* captures how much content an article contains (with the expectation that articles that do not contain much content are usually of low quality). The *number of references*, *number of links to other Wikipedia pages*, and *number of citation templates* show how the article editors support their content by using information from different sources, making the article more reliable and of higher quality. The *number of level 2 and level 3+ headings* reflects how the content is organized. Usually, Wikipedia articles of high quality have appropriate *number of level 2 and level 3+ headings.*

Readability scores reflect the use of language and how easy an article is to read. *Flesch reading score*, *Flesch-Kincaid grade level*, *Smog index*, and *Linsear write formula* use the average syllable per word or the number of polysyllables with different weight values to measure how difficult a text is to understand. Both *Coleman-Liau index* and *Automated readability index* use the average word length with different weight values to measure the readability of texts. *Readability consensus*, which is the estimated average school grade level required to understand the content, is computed by averaging across all readability scores in Table 7 except for *difficult words. Difficult words*, *Dale-Chall score*, and *Gunning-Fog index* use the number of difficult words or percentage of difficult words to measure the comprehension difficulty of a text. Here, a word is considered to be difficult if it is not in a list of 3000 common English words that fourth-grade American students can reliably understand.

### 6.2 Experimental Settings

In the multi-task learning scenario, we train a single model for each feature, i.e., learn 21 different models, one for each feature. The settings of biLSTM and Inception are the

| | *Length* | *References* | *Pagelinks* | *Citation* | *Non-citation* | *Categories* | *Images* |
|---|---|---|---|---|---|---|---|
| biLSTM | 0.8945 | 0.8542 | 0.8486 | 0.8397 | **0.8256**† | 0.6901 | 0.3278 |
| Inception | **0.9288**† | **0.8573** | **0.8861**† | **0.8764**† | 0.8036 | **0.8804**† | **0.8358**† |
| | *Infobox* | *Level2* | *Level3+* | *Flesch* | *Flesch-Kincaid* | *Smog* | *Coleman-Liau* |
| biLSTM | 0.8530 | 0.7211 | 0.7460 | **0.6209**† | **0.5488**† | **0.5368**† | 0.6646 |
| Inception | **0.8954**† | **0.8769**† | **0.9311**† | 0.5580 | 0.4602 | 0.3730 | **0.6685** |
| | *Readability Index* | *Difficult Words* | *Dale-Chall* | *Linsear* | *Gunning-Fog* | *Infonoise* | *Consensus* |
| biLSTM | 0.6163 | 0.9330 | **0.7351**† | **0.3238**† | **0.6270**† | **0.8273**† | **0.2212**† |
| Inception | **0.6379**† | **0.9765**† | 0.6732 | 0.2755 | 0.5248 | 0.8199 | 0.1870 |

Table 8: Pearson correlation ($r$) for each predicted feature, for biLSTM and Inception. The best result for each feature is indicated in bold, and marked with "†" if the difference is statistically significant (based on a one-tailed Wilcoxon signed-rank test; $p < 0.05$).

same as previously, except that the loss is computed differently. With the sole exception of *Infobox*, the features are continuous, and so are modeled as regression tasks. The combined loss is thus the (unweighted) sum of the document quality cross-entropy (between actual quality and predicted quality) and feature-level mean squared error (between the actual and predicted feature value). In each case, we standardize the feature by subtracting the mean and scaling to unit variance. In the case of *Infobox*, the target variable is binary (i.e., does the article contain an infobox or not), and so the combined loss is simply the sum of the cross-entropy for the document quality label and *Infobox* feature value.

### 6.3 Experimental Results

Table 8 summarizes the experimental results for the feature learning task, in terms of the Pearson correlation (all features other than *Infobox*) or accuracy (*Infobox*) at predicting each feature. Predictably, Inception outperforms biLSTM for all structural features, with the one exception of *Non-citation*, and the results achieved by Inception are significantly better for 8 out of 10 structural features. Also predictably, Inception achieves a much better performance at learning *Images* than biLSTM. We can observe that biLSTM is superior to Inception at learning readability scores, where the results achieved by biLSTM are statistically better than those of Inception for 8 out of 11 readability scores. Both biLSTM and Inception achieve better performance at learning structural features than learning readability scores except for *Difficult Words* and *Infonoise*. The higher correlation in *Difficult Words* and *Infonoise* can be explained by the fact that they are highly correlated with article length, which is easy to learn for both biLSTM and Inception. On the other hand, both biLSTM and Inception find it difficult to learn *Linsear* and *Consensus*, as these two features are based on syllable counts or the average of multiple readability scores, which are more subtle to model.

Table 9 provides the results for predicting document quality in the multi-task setting. We can observe that the performance of biLSTM is slightly improved by explicitly learning hand-crafted features and document quality simultaneously for 14 out of 21 features, while the performance of Inception benefits less from multi-task learning (only 2 features help in predicting document quality). Surprisingly, some features (such as *length*) harm the

| | *Length* | *References* | *Pagelinks* | *Citation* | *Non-citation* | *Categories* | *Images* |
|---|---|---|---|---|---|---|---|
| biLSTM | 53.9% | **54.3%** | **54.4%** | **54.6%** | 54.0% | 54.0% | 53.8% |
| Inception | 56.6% | 56.9% | **57.1%** | 56.8% | 56.9% | 56.7% | 56.6% |
| | *Infobox* | *Level2* | *Level3+* | *Flesch* | *Flesch-Kincaid* | *Smog* | *Coleman-Liau* |
| biLSTM | **54.3%** | **54.6%**$^{\dagger}$ | **54.2%** | **54.2%** | **54.2%** | **54.4%** | 54.0% |
| Inception | **57.2%** | 56.8% | 56.4% | 57.0% | 57.0% | 56.9% | 56.5% |
| | *Readability Index* | *Difficult Words* | *Dale-Chall* | *Linsear* | *Gunning-Fog* | *Infonoise* | *Consensus* |
| biLSTM | **54.2%** | 54.1% | 54.1% | **54.4%** | **54.2%** | **54.2%** | **54.4%** |
| Inception | 56.7% | 56.9% | 56.5% | 56.7% | 56.9% | 56.5% | 56.9% |

Table 9: Experimental results for document quality prediction under the multi-task setting. Results achieved by biLSTM and Inception in the multi-task setting which are better than those for the corresponding single task setting are indicated in bold, and marked with "†" if the difference is statistically significant (based on a one-tailed Wilcoxon signed-rank test; $p < 0.05$).

performance of both biLSTM and Inception. Examining the Pearson correlation between each feature and the document quality, we observe no relation between the correlation figures and performance over document quality assessment. For example, the correlation between *Length* and the quality label is 0.61, whereas *Length* harmed the performance of both biLSTM and Inception; the correlation between *Smog* and the quality labels is $-0.10$, and yet *Smog* improved the performance of biLSTM and compromised the performance of Inception.

### 6.4 Experiments Incorporating Hand-Crafted Features

Observing that models are not able to learn hand-crafted features fully, we propose to combine network learned features with hand-crafted features, as in Shen et al. (2017). To concatenate hand-crafted features with features learned by our models, we first normalize the hand-crafted features as above for Wikipedia (by subtracting the mean and scaling to unit variance, for each feature) and by min–max normalization for arXiv.[5] It is worth mentioning that we incorporate 21 dataset-specific features in Table 7 for Wikipedia, and 14 dataset-specific features for arXiv. As explained in CVPG section, extracting hand-crafted features is non-trivial for CVPG, which is the reason we don't report such results for CVPG.

Table 10 presents a comparison of results over Wikipedia and arXiv for our basic models, and models using hand-crafted features as side information. We can see that biLSTM$^{+}$, Inception$^{+}$, and Joint$^{+}$ achieve better results than their corresponding base models over Wikipedia and 2 out of 3 subsets of arXiv. Specifically, all models achieve statistically significant improvements over Wikipedia, and slightly better results (not at a level of significance) over 2 out of 3 of the arXiv subsets. We hypothesise that the hand-crafted features for Wikipedia are more powerful in capturing document quality than those for arXiv, in large part because of the editorial guidelines associated with Wikipedia documents, as compared to arXiv papers where writing styles and document structures vary much more.

[5] The decision to use min–max normalization for arXiv was a purely empirical one, in that it led to better results, whereas in the Wikipedia case, $z$-scoring performed better.

| | | biLSTM | biLSTM$^+$ | Inception | Inception$^+$ | Joint | Joint$^+$ |
|---|---|---|---|---|---|---|---|
| Wikipedia | | 54.1±0.47% | **57.2±0.45%**$^\dagger$ | 57.0±0.63% | **58.8±0.71%**$^\dagger$ | 59.4±0.47% | **62.5±0.51%**$^\dagger$ |
| arXiv | cs.ai | 91.5±1.03% | **92.1±1.06%** | 92.8±0.79% | 92.8±0.81% | 93.4±1.07% | 93.1±0.95% |
| | cs.cl | 76.2±1.30% | **76.8±1.67%** | 76.2±2.92% | **76.7±2.09%** | 77.1±3.10% | **77.2±2.59%** |
| | cs.lg | 81.1±0.83% | 80.0±2.30% | 79.3±2.94% | **79.5±1.28%** | 79.9±2.54% | **81.2±1.05%** |

Table 10: Experimental results incorporating the hand-crafted features. biLSTM$^+$, Inception$^+$, and Joint$^+$ are the models that concatenate hand-crafted features with features learned by biLSTM, Inception, and Joint. Results achieved by biLSTM$^+$, Inception$^+$, and Joint$^+$ are indicated in **bold**, if they are better than those achieved by biLSTM, Inception, and Joint, respectively, and additionally with "†" if the difference is statistically better (based on a one-tailed Wilcoxon signed-rank test; $p < 0.05$).

## 7. Conclusion

In this paper, we proposed to use visual renderings of documents to capture implicit document quality indicators, such as font choices, images, and visual layout, which are not captured in textual content. We applied neural network models to capture visual features given visual renderings of documents. Experimental results show that we achieve at least 2.2% higher accuracy than state-of-the-art approaches based on textual features over Wikipedia, and performance competitive with or surpassing state-of-the-art approaches over arXiv. We further proposed a joint model, combining textual and visual representations, to predict the quality of a document. Experimental results show that our joint model outperforms the visual-only model in all cases and the text-only model in all cases except for cs.lg. These results underline the feasibility of assessing document quality via visual features, and the complementarity of visual and textual document representations for quality assessment. This also demonstrates the general applicability of our proposed model. In the multi-task setting, we observe that our visual model outperforms the textual model in learning structural features, while the textual model is better at learning readability scores, further verifying the complementary of textual and visual features. In future work, we intend to not only predict the quality of Wikipedia articles but also provide feedback to end users as to which parts of the article need improvement. We also intend to apply our joint model to other domains, such as cQA and essay scoring.

## References

Agichtein, E., Castillo, C., Donato, D., Gionis, A., & Mishne, G. (2008). Finding high-quality content in social media. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, pp. 183–194.

Alikaniotis, D., Yannakoudakis, H., & Rei, M. (2016). Automatic text scoring using neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 715–725.

Attali, Y., & Burstein, J. (2004). Automated essay scoring with e-rater® v.2.0. *The Journal of Technology, Learning and Assessment*, *4*(3), 1–30.

Belinkov, Y., Mohtarami, M., Cyphers, S., & Glass, J. R. (2015). VectorSLU: A continuous word vector approach to answer selection in community question answering systems. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pp. 282–287.

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research*, *3*, 1137–1155.

Bird, S. (2006). NLTK: The Natural Language Toolkit. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pp. 69–72.

Bruni, E., Tran, N., & Baroni, M. (2014). Multimodal distributional semantics. *Journal of Artificial Intelligence Research*, *49*, 1–47.

Chall, J. S., & Dale, E. (1995). *Readability revisited: The new Dale-Chall readability formula*. Brookline Books.

Chen, H.-H. (2012). How to use readability formulas to access and select English reading materials. *Journal of Educational Media & Library Sciences*, *50*(2), 229–254.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734.

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1800–1807.

Coleman, M., & Liau, T. L. (1975). A computer readability formula designed for machine scoring. *Journal of Applied Psychology*, *60*(2), 283–284.

Cummins, R., Zhang, M., & Briscoe, T. (2016). Constrained multi-task learning for automated essay scoring. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 789–799.

Dale, E., & Chall, J. S. (1948). A formula for predicting readability: Instructions. *Educational Research Bulletin*, *27*(1), 37–54.

Dalip, D. H., Gonçalves, M. A., Cristo, M., & Calado, P. (2017). A general multiview framework for assessing the quality of collaboratively created content on web 2.0. *Journal of the Association for Information Science and Technology*, *68*(2), 286–308.

Dalip, D. H., Lima, H., Gonçalves, M. A., Cristo, M., & Calado, P. (2014). Quality assessment of collaborative content with minimal information. In *Proceedings of the 14th ACM/IEEE-CS Joint Conference on Digital Libraries*, pp. 201–210.

Dang, Q.-V., & Ignat, C.-L. (2016a). Measuring quality of collaboratively edited documents: The case of Wikipedia. In *Proceedings of the 2nd IEEE International Conference on Collaboration and Internet Computing*, pp. 266–275.

Dang, Q.-V., & Ignat, C.-L. (2016b). Quality assessment of Wikipedia articles without feature engineering. In *Proceedings of the 16th ACM/IEEE-CS on Joint Conference on Digital Libraries*, pp. 27–30.

Dang, Q.-V., & Ignat, C.-L. (2017). An end-to-end learning solution for assessing the quality of Wikipedia articles. In *Proceedings of the 13th International Symposium on Open Collaboration*, pp. 4:1–4:10.

Dong, F., Zhang, Y., & Yang, J. (2017). Attention-based recurrent convolutional neural network for automatic essay scoring. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pp. 153–162.

Gunning, R. (1969). The Fog index after twenty years. *International Journal of Business Communication*, *6*(2), 3–13.

Guzmán, F., Màrquez, L., & Nakov, P. (2016a). Machine translation evaluation meets community question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 460–466.

Guzmán, F., Nakov, P., & Màrquez, L. (2016b). MTE-NN at SemEval-2016 task 3: Can machine translation evaluation help community question answering?. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pp. 887–895.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Hoogeveen, D., Wang, L., Baldwin, T., & Verspoor, K. M. (2018). Web forum retrieval and text analytics: A survey. *Foundations and Trends in Information Retrieval*, *12*(1), 1–163.

Hou, Y., Tan, C., Wang, X., Zhang, Y., Xu, J., & Chen, Q. (2015). HITSZ-ICRC: Exploiting classification approach for answer selection in community question answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pp. 196–202.

Huang, J. (2018). Deep paper gestalt. *arXiv preprint arXiv:1812.08775.*

ImageMagick (2020). A tool to convert pdf to screenshot. `https://www.imagemagick.org/script/index.php`.

ImageNet (2020). Imagenet dataset. `https://www.image-net.org`.

Jeon, J., Croft, W. B., Lee, J. H., & Park, S. (2006). A framework to predict the quality of answers with non-textual features. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 228–235.

Kang, D., Ammar, W., Dalvi, B., van Zuylen, M., Kohlmeier, S., Hovy, E. H., & Schwartz, R. (2018). A dataset of peer reviews (PeerRead): Collection, insights and NLP applications. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1647–1661.

Kincaid, J. P., Fishburne Jr., R. P., Rogers, R. L., & Chissom, B. S. (1975). Derivation of new readability formulas (Automated Readability Index, Fog Count and Flesch Reading Ease formula) for Navy enlisted personnel. Tech. rep., DTIC Document.

Kingma, D., & Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*.

Lau, J. H., & Baldwin, T. (2016). An empirical evaluation of doc2vec with practical insights into document embedding generation. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pp. 78–86.

Le, Q. V., & Mikolov, T. (2014). Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning*, pp. 1188–1196.

Lipka, N., & Stein, B. (2010). Identifying featured articles in Wikipedia: Writing style matters. In *Proceedings of the 19th International Conference on World Wide Web*, pp. 1147–1148.

McLaughlin, G. H. (1969). SMOG grading: A new readability formula. *Journal of Reading*, *12*(8), 639–646.

Parse (2020). Science parse library. `https://github.com/allenai/science-parse`.

Pennington, J., Socher, R., & Manning, C. (2014). GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.

Persing, I., & Ng, V. (2013). Modeling thesis clarity in student essays. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 260–269.

Persing, I., & Ng, V. (2014). Modeling prompt adherence in student essays. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1534–1543.

Phandi, P., Chai, K. M. A., & Ng, H. T. (2015). Flexible domain adaptation for automated essay scoring using correlated linear regression. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 431–439.

PyPDF2 (2020). Python package to trim and pad pdf documents. `https://pypi.org/project/PyPDF2/`.

Rendering (2020). Wikipedia visual rendering script. `https://github.com/ariya/phantomjs/blob/master/examples/rasterize.js`.

Scheme (2020). Wikipedia grading scheme. `https://en.wikipedia.org/wiki/Template:Grading_scheme`.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the 2017 IEEE International Conference on Computer Vision*, pp. 618–626.

Shen, A., Qi, J., & Baldwin, T. (2017). A hybrid model for quality assessment of Wikipedia articles. In *Proceedings of the Australasian Language Technology Association Workshop*, pp. 43–52.

Shen, A., Salehi, B., Baldwin, T., & Qi, J. (2019). A joint model for multimodal document quality assessment. In *Proceedings of the 18th Joint Conference on Digital Libraries*.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *Proceedings of the 3rd International Conference on Learning Representations*.

Smith, E. A., & Senter, R. J. (1967). Automated readability index. *Aerospace Medical Research Laboratories (U.S.)*, 1–14.

Statistics (2020). Wikipedia statistics. `https://en.wikipedia.org/wiki/Wikipedia:Statistics`.

Stein, K., & Hess, C. (2007). Does it matter who contributes: A study on featured articles in the German Wikipedia. In *Proceedings of the Eighteenth Conference on Hypertext and Hypermedia*, pp. 171–174.

Suggu, S. P., Goutham, K. N., Chinnakotla, M. K., & Shrivastava, M. (2016). Hand in glove: Deep feature fusion network architectures for answer quality prediction in community question answering. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 1429–1440.

Suryanto, M. A., Lim, E., Sun, A., & Chiang, R. H. L. (2009). Quality-aware collaborative question answering: Methods and evaluation. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pp. 142–151.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826.

Taghipour, K., & Ng, H. T. (2016). A neural approach to automated essay scoring. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1882–1891.

Talk (2020). Wikipedia talk pages. `https://en.wikipedia.org/wiki/Help:Talk_pages`.

Tang, D., Qin, B., & Liu, T. (2015). Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1422–1432.

Tay, Y., Phan, M. C., Tuan, L. A., & Hui, S. C. (2018). SkipFlow: Incorporating neural coherence features for end-to-end automatic text scoring. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, pp. 5948–5955.

TextStat (2020). Textstat. `https://pypi.python.org/pypi/textstat/0.5.1`.

van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Rearch*, *9*, 2579–2605.

Wang, P., & Li, X. (2020). Assessing the quality of information on Wikipedia: A deep-learning approach. *Journal of the Association for Information Science and Technology*, *71*(1), 16–28.

Wang, S., & Iwaihara, M. (2011). Quality evaluation of Wikipedia articles through edit history and editor groups. In *Proceedings of the 13th Asia-Pacific Web Conference on Web Technologies and Applications*, pp. 188–199.

Warncke-Wang, M., Ayukaev, V. R., Hecht, B., & Terveen, L. (2015). The success and failure of quality improvement projects in peer production communities. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, pp. 743–756.

Warncke-Wang, M., Cosley, D., & Riedl, J. (2013). Tell me more: An actionable quality model for Wikipedia. In *Proceedings of the 9th International Symposium on Open Collaboration*, pp. 8:1–8:10.

WikiClass (2020). Wikiclass. `https://github.com/wiki-ai/wikiclass`.

Yang, P., Sun, X., Li, W., & Ma, S. (2018). Automatic academic paper rating based on modularized hierarchical convolutional neural network. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 496–502.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., & Hovy, E. (2016). Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1480–1489.

Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Proceedings of the Twenty-ninth Conference on Neural Information Processing Systems*, pp. 649–657.

Zhou, X., Hu, B., Chen, Q., Tang, B., & Wang, X. (2015). Answer sequence learning with neural networks for answer selection in community question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pp. 713–718.

Zhu, X., & Gauch, S. (2000). Incorporating quality metrics in centralized/distributed information retrieval on the world wide web. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 288–295.