Enabling text readability awareness during the micro planning phase of NLG applications[1]

Priscilla Moraes, Kathleen McCoy, Sandra Carberry
Computer and Information Sciences Department
University of Delaware, Newark, DE
pmoraes | mccoy | carberry@udel.edu

Abstract

Currently, there is a lack of text complexity awareness in NLG systems. Much attention has been given to text simplification. However, based upon results of an experiment, we unveiled that sophisticated readers in fact would rather read more sophisticated text, instead of the simplest text they could get. Therefore, we propose a technique that considers different readability levels during the micro planning phase of an NLG system. Our technique considers grammatical and syntactic choices, as well as lexical items, when generating text. The application uses the domain of descriptive summaries of line graphs as its use case. The technique proposed uses learning for identifying features of text complexity; a graph search algorithm for efficient aggregation given a target reading level, and a combination of language modeling and word vectors for the creation of a domain-aware *synset* which allows the creation of disambiguated lexicon that is appropriate to different reading levels. We found that generating text at different target reading levels is indeed preferred by readers with varying reading abilities. To the best of our knowledge, this is the first time readability awareness is considered in the micro planning phase of NLG systems.

1 Introduction

Prior work has concentrated on simplifying text in order to make it accessible to people with cognitive disabilities or low literacy levels. On the other hand, as stated by (Williams & Reiter, 2005b), most NLG systems generate text for readers with good reading ability. Our contention, however, is that NLG systems will be much more effective if they can target their output to the preferences of the reader. This not only enables easy comprehension, but it also makes the experience more enjoyable for them. Based on that claim, we propose an approach that considers a target reading level in order to decide on the syntactic and grammatical structure of the generated text and to select appropriate lexical items.

Our overall goal is to generate text at a target reading level. The process of generating text takes in a number of propositions and outputs a set of English sentences which are realizations of these propositions. Each proposition can be realized in several different ways, e.g., as a single sentence, aggregated with another proposition as an adjective attached to a noun, as a relative clause, or as another noun phrase in a coordination. In addition, different lexical items can be used to describe a term, and these might also vary in complexity and grade level appropriateness. The devised approach is applied to the domain of line graph description. Information graphics (non-pictorial images such as line graphs, bar and pie charts) are commonly used by authors in

[1] This document has been adapted from the instructions for earlier ACL and NAACL proceedings, including those for NAACL-HLT-15 by Matt Post and Adam Lopez, NAACL-HLT-12 by Nizar Habash and William Schuler, NAACL-HLT-10 by Claudia Leacock and Richard Wicen- towski, NAACL-HLT-09 by Joakim Nivre and Noah Smith, for ACL-05 by Hwee Tou Ng and Kemal Oflazer, for ACL-02 by Eugene Charniak and Dekang Lin, and earlier ACL and EACL formats. Those versions were written by several people, including John Chen, Henry S. Thompson and Donald Walker. Additional elements were taken from the formatting instructions of the *International Joint Conference on Artificial Intelligence*. Microsoft Word formatting was added by Alexander Mamishev (Mamishev, 2013).

Proceedings of The 9th International Natural Language Generation conference, pages 121–131,
Edinburgh, UK, September 5-8 2016.

order to convey a message or to make a point regarding the topic being discussed in the document or article.

To efficiently select realizations at a particular reading level, we devised an approach that uses a graph search algorithm guided by a heuristic. To construct the heuristic, the features of text complexity were identified through machine learning. The lexical choice implements a concept expansion phase followed by an approach which combines language modeling and word vectors to disambiguate domain-relevant concepts. In the last step a grade level appropriate lexicon is applied. To the best of our knowledge, this is the first effort made during the micro planning phase of an NLG system that both **considers different reading abilities when generating text** and **presents automated approaches in order to do it**.

The next section describes related work on text readability and text simplification. Sections 3 to 5 discuss the graph search algorithm for the aggregation phase, the learning of feature measurements and the identification of grade level appropriate and domain aware lexicons. Section 6 shows some examples of summaries generated by the system. Section 7 presents the evaluations of the system and Sections 8 and 9 provide conclusions and thoughts on future work, respectively.

2 Related Work

The approach proposed by (Wilkinson, 1995) presents the aggregation process divided into two major steps: semantic grouping and sentence structuring. Although they are interdependent, both are needed in order to achieve aggregation in a text. (Barzilay & Lapata, 2006), (Bayyarapu, 2011), (Walker, Rambow, & Rogati, 2001) are some examples of learning aggregation rules and grouping constraints in order to aggregate text. It differs from our approach in that we are considering readability constraints when making such decisions.

(Elhadad, Robin, & McKeown, 1997) present work on lexical choice considering constraints regarding syntax, semantics, pragmatics, the lexicon, and the underlying domain that float from one phase to the next in the generation of text. Our work differs in that lexical items are restrained by their appropriateness to the level of the reader and the creation of the lexicon is guided by defining a domain-aware *synset* for description of line graphs.

Other NLG systems decide on text complexity based on available scales such as the D-level sentence complexity (Covington, He, Brown, Naci, & Brown, 2006). One example is presented in (Demir, Carberry, & McCoy, 2012), where tree structures are built representing all the possible ways sentences can be aggregated and the choice of the tree attempts to balance the number of sentences, their D-level complexity, and the types of relative clauses. The work presented in (P. Moraes, McCoy, & Carberry, 2014) describe a template-based approach for creating summaries at different reading levels. It does not, however, present an adaptive approach that can be applied to the micro planning phase of any NLG system.

Another area, text simplification, aims to target low-skilled readers and users with language disabilities. SkillSum (Williams & Reiter, 2004, 2005a; Williams, Reiter, & Osman, 2003) is a system which adapts its output for readers with poor literacy after assessing their reading and numeracy skills. Their results show that, for these target readers, the micro planning choices made by SkillSum enhanced readability. (Carroll et al., 1999) presents a text simplification methodology to help language-impaired users; (Rello, Baeza-Yates, Bott, & Saggion, 2013) propose a system that uses lexical simplification to enhance readability and understandability of text for people with dyslexia; while (Siddharthan, 2003) aims to make the text easier to read for some target group (like aphasics and people with low reading ages) or easier to process by some program (like a parser or machine translation system). One of our evaluation experiments (citation suppressed for anonymity) performed with college students showed that the simplest text was rather unpleasant for them to read. We therefore propose a technique that focuses on adjusting the generated text to the reading level of the surrounding text.

The closest work to the one proposed in this paper is presented in (Bateman & Paris, 1989). It p resents an approach to tailoring phrasing during the generation of natural text to different types of users. It employs a technique that leverages a knowledge base in order to make decisions during text planning in a rule based fashion. This work, in contrast, generates natural text aimed at a specific reading level

by applying a graph search that allows the automation of the aggregation of propositions.

3 Aggregation of Propositions

The goal of the micro planning phase in NLG systems is to realize the set of selected propositions as sentences. The NLG system developed in the context of this work generates summaries of the high-level message conveyed by line graphs present in popular media. In the context of describing line graphs, there are many ways these propositions can be realized. They can each constitute a sentence; some of them can be realized as an adjective attached to a noun phrase, as a noun phrase added to a conjunction with a preexisting noun phrase, or as a subordinating conjunction. The last three realization options require what we call aggregation of propositions, where multiple propositions are composed to form a complete sentence. Consider how the proposition **graph_type**, for example, can be realized: A sentence: "There is a line graph." / An adjective (or compound noun): "…line graph…" – where "graph" is the head noun / A relative clause: "…which is lined…" – where the head noun is "graph".

The other propositions in the context of line graphs can also have their realizations made in different ways (based on grammatical restrictions on how each concept can be described) and the realizations constrain each other. A hard decision, therefore, is to choose which realization to apply to each proposition. We decided to implement the aggregation phase by employing a graph search algorithm. Since there is a large number of options and we do not know which combination will give us a final summary at the desired reading level, a graph search allows us to explore the whole search space and, through the use of a heuristic, to efficiently get to a goal node.

3.1 The Graph Search Problem

The search space for the aggregation of propositions problem is defined as:

States: A state consists of two parts: a list of unrealized propositions and the realizations performed so far (which can consist of full sentences or sentence fragments). **Initial state**: The initial state contains the set of all unrealized propositions. **Goal state**: It checks if all the propositions have been realized and if all of them are aggregated into full sentences. **Actions**: The actions in a given state take the next unrealized proposition and realize it (generating a new state for each realization the proposition allows). For most propositions, the possible actions are: *realize_as_active_sentence, realize_as_passive_sentence, realize_as_adjective, realize_as_relative_clause and realize_as_conjunction*. Each proposition contains a set of its allowed actions. When realizing a proposition as a fragment, if the needed head noun is not present in any of the realizations, then the proposition will be realized as the respective fragment (adjective, relative clause, conjunction) and will wait until such a head noun is generated to be added to a full sentence. If the required head noun is already realized in a full sentence, the fragment is then attached to the existing realization.

In order to find a goal node efficiently, we developed a heuristic that takes into account three factors. The first factor considers the realizations performed so far in a node. It measures the values of the different features of text complexity (explained in the next section) in the summary realized so far. The second factor estimates how likely a node is to stay within a range of allowed values for the different features that define text complexity. This estimation is done by looking at the propositions that still need to be realized and the probability of them increasing or decreasing the values of the features. The third factor favors nodes that are deeper in the tree. Since all goal nodes are at the same level (defined by the number of unrealized propositions), this aspect favors the nodes that are closer to a goal.

To be able to build the heuristic, the set of features of text complexity that could be explicitly measured and used during generation had to be identified. For that, we use a learning approach that classifies text into different target reading levels and that provides the values that the features had in the different classifications. The next section explains the learning approach.

4 Learning Text Complexity Features

The features to be used in the heuristic needed to be chosen based on both their effect on text complexity and their usability. The choice of features for constructing the model was made based on the work

presented by (Vajjala & Meurers, 2012) which uses features that are based on Second Language Acquisition (SLA) research combined with traditional readability features, such as word length and sentence length, in order to classify text into different grades. Their work results in classifiers that outperform previous approaches on readability classification, reaching higher classification accuracy. However, since we still need to map features back to the NLG aggregation phase, the set of features used here represents a subset of the features presented in their work. The final set of features, motivated by (Vajjala & Meurers, 2012), consisted of 15 features. Examples of features are: *Percentage of passive sentences (percPassiveSent); Percentage of conjunctions (percConjunction); Percentage of prepositions (percPreposition); Percentage of adjectives (percAdjective); Percentage of adverbs (percAdverb); Percentage of relative clauses (percRelativeClauses); Average noun phrase length (avgNounPhraseLength); Average sentence length (avgSentLengthWord)*;

For the learning algorithm a decision tree is used. The goal of the learning algorithm was to provide the system with concrete measures of the chosen features that can be mapped to the graph search heuristic during the aggregation phase.

4.1 Corpus of Grade Level Annotated Text

Data was obtained from text exemplars classified at different grade bands available in Appendix B of the Common Core State Standards (Common Core State Standards Initiative, 2010) and various articles written and annotated at different reading levels. Magazine articles collected from the Austin Public Library electronic catalog (Library, 2015) were annotated using the Lexile measure ("Lexile Framework for Reading," 2015). Classes for the learning algorithm were grouped as **4th - 5th grades**, **6th - 8th**, **9th - 10th**, and **11th - College**. One hundred articles, varying in size, were collected for each one of the grade level groups. These articles were in HTML format and they were preprocessed to remove tags and special characters. After preprocessing the files, they were split into smaller passages, of at least 150 words, which is equivalent to the average size of the summaries the system generates. Because the passages needed to have complete sentences in order to obtain more accurate measurement of the features during learning, the splitting step counted words sentence by sentence and, after reaching 150 words, it stopped adding sentences to the current passage. Splitting the articles resulted in 1874 passages, which were used as instances in the learning algorithm.

After splitting the articles into similar passage sizes, the values of the features were calculated using the Style & Diction tool (FSF, 2005) for assessing some of the syntactic features and NLTK (Loper & Bird, 2002) for grammatical features. After all the features were assessed, a tab file (appropriate input file type for use with the Orange toolbox (Demsar et al., 2013) is generated and ready for training.

4.2 Classification Task

Before choosing decision trees as the learning algorithm to be used for this classification task, other algorithms were analyzed using the data described in the previous section and their results were compared. Random forests, Bayesian networks, Classification (or decision) trees and Neural Networks were applied to the classification task. Using leave-one-out cross validation, the system achieved a classification accuracy of 85.38% and F1 measure of 87.97% using decision trees. The Neural Network outperformed the classification accuracy of the decision tree by 1.39%, but had a smaller F1 measure. The neural network used 20 hidden layers, which would probably complicate reading the features weights due to the combination functions within the hidden layers. Since the goal is to be able to map the weights of the features to a heuristic in a graph search algorithm, the best option turned out to be the decision tree since it can be interpreted as rules, which allow the values of the features to be captured.

The paths from the root to the leaves (or classes, in this case) provide logical rules that represent the values of the different features that led to that classification. The logic rules can be read as *path1 OR path2 OR ... pathN* for a given grade level group (grade level groups are the target classes of the leaf nodes). Only nodes with a classification confidence above 70 percent were used to construct the set of logic rules that is used by the system. A set of rules for a 9th – 10th grade level band is shown here as an example of what the decision tree produces:

(avgParagLengthSent <= 10 AND
(13 < avgSentLengthWord <= 15 AND
percPassiveSent <= 0.4 AND
percRelativeClauses<=0.6AND
0.2 < percBegSentPronoun <= 0.5)
OR
(avgParagLengthSent <= 9 AND
(14 < avgSentLengthWord <= 16) AND
percPassiveSent <= 0.1 AND
percRelativeClauses <= 0.8)

We use rules such as this to build the heuristic to help guide the search to a realization that satisfies the target reading level. When using these rules within our heuristic, the function will be estimating the cost based on how well the to-be-realized propositions, combined with the realizations performed so far, fall within those ranges in order to be inside the grade level constraints.

4.3 Mapping the Rules to a Heuristic Function

In calculating the heuristic, for the propositions that have not been realized yet, the features are divided into two groups. The first group contains features that only increase as new propositions are realized. One example is the number of relative clauses in a paragraph. As the number of sentences in the paragraph increases, the value of this feature can never go down. The second group contains features whose values can fluctuate (either up or down) as new propositions are realized. The average sentence length in words, for example, can go up or down as new propositions are realized since they can become new sentences (making it go down) or be aggregated with existing sentences (making it go up). For this reason, the heuristic calculates the estimated cost that is added to h(n) differently for these two groups.

Estimating the Cost Added by Feature Values

To illustrate, suppose that the decision tree learned that, for paragraphs that contain around 150 words, the range of values for the *numberAdjectives* feature is 2 <= *numberAdjectives* <= 5 for a 4th grade level text. The sequence of rules to calculate the cost for this type of feature is:

1. If the measured value of the feature in what has already been realized is above the upper limit of its range (if it is equal to 6 for the example above), add an infinite cost to the estimation. Since these feature's values can never go down, this node cannot satisfy the requirements for the grade level.
2. If the measured feature is within the predefined range (if it is equal to 3 for the example above), add to the estimation the probability of increasing the value of the feature based on the unrealized propositions. In this case, the probability of increasing the feature is the ratio of possible realizations that increase the feature's value (e.g. a proposition that has a possible realization as an adjective will increase the *numberAdjectives*) over all possible realizations amongst the set of unrealized propositions. In the example above, if there were 6 unrealized propositions from which 2 could be realized as active and passive voice sentence (4 possible realizations), 1 could be realized as active voice, passive voice sentence and relative clause (3 possible realizations), and 3 could be realized as active voice sentence, passive voice sentence, adjective, and relative clause (12 possible realizations), the number of possible realizations would be 19. Since only 3 could be realized as an adjective, the probability of increasing the value of this feature is 3/19 (~ 0.16). This value would be added to the cost, versus 0.31 (6/19) if there were 6 possible realizations as adjectives in the set of all possible realizations.
3. If the measured value is less than the lower limit (if it is equal to 1 for the example above), multiply the probability of increasing the value of the feature given the unrealized propositions (as explained above) by the inverse of the value that the feature can increase by (feature upper limit – feature value = 2 for the example above), then multiply the result by the number of possible realizations that use the feature. In this case, the more chances to realize a proposition as an adjective the better since the value is currently lower than desired.

Following the same logic, the calculation of features that fluctuate is performed by also taking into account the fact that the feature values can also fall under the lower limit provided by the rules (a case which the cost estimation also needs to address).

The final value for *h(n)* is the sum of all estimated costs when going through the set of features defined by the rules. The node with lowest value is expanded next.

5 Lexical Choice for Generating Summaries at Different Grade Levels

The lexicalization phase of this work is composed of three main sub phases. The first is a concept expansion phase achieved by the collection of synonyms starting from a set of seed words used to describe the different concepts of line graphs. The second step is concerned with narrowing the set of synonyms to the ones that are relevant to the domain of line graphs. This disambiguation step is performed by using language modeling (5-grams from Google Books) (Michel et al., 2011) and word vectors (word2vec) (Mikolov, Chen, Corrado, & Dean, 2013). The last step builds lexicons, based on the final set of synonyms for a concept, which are appropriate to the different target reading levels.

The seed words (base lexical item for each one of the concepts) were gathered from an experiment performed by (Greenbacker, Carberry, & McCoy, 2011) in which participants were asked to describe the important aspects they noticed were present in line graphs. From these passages, the most common words used to describe concepts such as volatility and steepness were used as the starting point for lexical building.

For expanding these concepts, Thesaurus.com (Dictionary.com, 2015) was used. Thesaurus.com was selected because it has a better coverage with respect to synonyms of nouns, verbs, adjectives and adverbs than WordNet (Fellbaum, 1998) and VerbNet (Kipper, Dang, & Palmer, 2000). Thesaurus.com provides synonyms for concepts in a varied number of senses and parts of speech by grouping synonyms within part_of_speech + *synsets*.

Choosing the most appropriate concept *synsets* for the domain of line graphs did not appear to be the best approach, as the *synsets* were not always comprehensive and precise. In other words, all *synsets* individually contained some synonyms which were not appropriate and appropriate synonyms were found across multiple *synsets*. Besides, choosing a single best *synset* would not lead to a technique that could perform the synonym expansion without human supervision. For this reason, the decision was therefore to use all *synsets* with a given part of speech and to further filter the resulting set.

This provided the system with an extensive (and noisy) list of synonyms. The set of synonyms was too broad; it included synonyms that would not apply to the domain of line graph description, so disambiguating the synonyms and filtering only the domain relevant ones was needed.

5.1 Using Language Modeling and Word Vectors for Filtering Synonyms

The intuition here is that we want to keep only synonyms that the language model indicates appear in a context containing key words indicative of the line graph context.

The language model used is the 5-gram corpus from Google Books (Michel et al., 2011). The system selects all the 5-gram instances that were found to contain a synonym of the concept being expanded which co-occurs with one of the words from the "concept context". The concept context is the set of head nouns that can appear in a sentence with the concept being expanded; in the example above, the concept context for "show" would be the terms "image", "graph", and "trend", since the possible contexts are the sentences: "The image shows a graph" and "The graph shows a trend". This set of lexical contexts is the same one used to seed the lexical expansion of concepts described earlier and originated from the most common terms used to express concepts in the experiment presented in (Greenbacker et al., 2011).

However, the set still contained terms that were inappropriate for the graph summarization domain. Thus, we devised a vector space model approach trained on Wikipedia (Wikipedia, 2004) data, which is available as a default corpus for training the word2vec tool available at (Mikolov et al., 2013). Word representation in vector spaces has shown to be a promising tool for acquiring terms' semantic knowledge. This technique builds vectors that represent the context of a term. The vector for the term "house", for example, has a higher count for the terms "big", "white", "spacious", than for the terms "hungry", "bag", and "sky". The vector is built by assigning co-occurrence counts to all the words in the language in question, and two terms can be compared on how similar they are in their contexts by measuring the similarity of their vectors. The idea is that two synonyms ought to occur in the same linguistic context; therefore, their word2vec scores should be very close.

By using the word2vec tool, the system was able to filter the set of synonyms collected from the language model step and further customize it to the line graph domain context. The reader might ask why both steps are needed in order to come up with the set of appropriate synonyms. It was noticed that the language model alone was not sufficient since no threshold could be set in the system in order to consider a synonym for inclusion in the set, the reason being that any threshold eliminated the chances of good synonyms for the context of line graphs (volatility, for example), that were not as commonly used in the literature, from being added to the set. Using word vector representations alone, on the other hand, poses another challenge. The approach used by vector representation does not allow differentiation of a synonym from an antonym. The words "pretty" and "ugly" would have a very similar vector representation since they can be used within the same context. By collecting synonyms from a dictionary and starting the set of possible replacements from them, the antonyms were already filtered. By filtering co-occurrence present in Google N-grams (generated from digitized books), the noise is significantly decreased. One can then perform additional filtering by looking at the vector space models of the senses being disambiguated, which has good results for the line graph use case. This combined approach proved to be a way of allowing a system to create a customized *synset* of a domain by starting from a set of context words.

5.2 Creation of Lexicons for Different Reading Levels

These disambiguation steps enabled the system to come up with a set of terms that were appropriate lexical items for the line graph concepts needed for our summaries. Since the focus of the system is to generate text at different grade levels, a step to bin those terms based on their grade level appropriateness was also necessary. For any given concept, some of the lexical items may be rather simple and others might be considered more advanced.

In order to build grade level appropriate lexicons, the final set of synonyms disambiguated for the line graph domain was further divided into grade levels by checking for their lemma forms in the data previously used to learn text complexity feature measurements (the annotated corpus of different grade levels). From this step, each group of grade levels ended up with one or more terms that could describe the concepts used to generate descriptive summaries of line graphs. Since lexical choice can affect the final readability measurement of the generated text, the system randomly selects terms at the target reading level that will represent concepts before starting the graph search explained earlier in the previous section. Evaluation results for the micro planning phase are presented in Section 7.

6 Summaries at Different Reading Levels

The following summaries illustrate the different output from the system given different target reading levels. These summaries were generated for the graph shown in Figure 1. The propositions used in the generated summaries were provided by the content selection module of the system, which employs a centrality-based algorithm in order to select propositions (P. S. Moraes, Carberry, & McCoy, 2013).

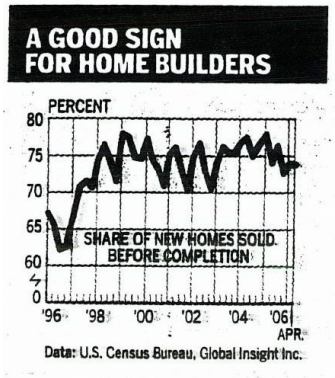

Figure 1: Example of graph extracted from online popular media.

4th – 5th summary: *There is an image. The image shows a line graph. The share of new homes sold before completion in percent is given by the graph. The graph consists of a changing trend composed of a rising trend from 1996 to 1999 followed by a stable trend through 2006. The graph is variable. The graph has the top value of 78.09 percent. The graph has the lowest value of 62.65 percent.*

11th – College summary: *A volatile line diagram, which presents the share of new homes sold before completion in percent and consists of a changing trend composed of a rising trend from 1996 to 1999 followed by a stable trend through 2006, is revealed by the image. The maximum value of 78.09 percent is reached by the graph, which has the minimal value of 62.65 percent.*

7 Evaluation of the Microplanning Phase

Four different evaluations were performed to assess the effectives of the system on generating summaries at different target reading levels. These evaluations intended to assess:

1) **The ability of the system, given a target reading level, to generate a summary that is as close as possible to that target**. For this experiment we used a set of 11 line graphs. We ran the system five times, generating five slightly different summaries at the reading level identified for the article in which the graphs appeared. These five summaries differ since, on each iteration of the system, the lexical choice randomly selects lexical items from the pool of appropriate options. The average grade level was used as the final reading level. 63.6% of the graphs had their summaries produced by the system matching their target reading level exactly. 27.3% of the graphs had their summaries generated by the system really close to the target reading level, having 1 summary produced at grade level 8.8 with a target 9^{th} – 10^{th} and 2 summaries produced at grade levels 10.7 and 10.9 with a target 11^{th} - College. And only one of the graphs had the summary generated at 2 grade levels lower than the target reading level (7.2 with a target of 9^{th} – 10^{th}).

2) **The ability of the system to generate different summaries appropriate for different grade levels for any given graph in the experiment set**. The system was able to successfully generate summaries for all 11 graphs with increasing complexity as the target reading level increased (this also used the average of five runs). It generated 11% of the summaries at a reading level that did not change, having generated summaries at 9^{th} - 10^{th} grade level that targeted the 11^{th} – college grade group. This was due to the lack of enough propositions to perform grammatical combinations that would lead to a higher reading level.

3) **The ability of the system on varying the text complexity as perceived by human readers**. For this experiment, 90 Human Intelligent Tasks (HITs) were undertaken through Amazon Mechanical Turk (10 graphs, 9 *turkers* per graph). Each HIT produced an ordering which corresponded to the grade level the *turkers* believed the summaries belonged to. Each summary could be associated to only one grade level. Since choosing one wrong grade level to a summary would lead to another misclassification, a pair-wise relationship approach was applied to analyze the results. From 348 valid pairwise relationship results, 252 had a correct ordering, yielding a similarity between human readers and the system's perception of text complexity of 72%. Another evaluation made on the results provided by the *turkers* was through calculating the average of the nDCGs obtained on the orderings. Using the formula presented in Figure 2, the results obtained were the ones presented in Table 1.

$$\mathrm{DCG_p} = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2(i)}$$

Figure 2: nDCG formula used to assess the goodness of the ordering.

Graph	nDCG
L3	0.9598
L6	0.9860
L18	0.8975
L21	0.8893
L23	0.9451
L26	0.9752
L28	0.9888
L42	0.9365
L89	0.9851
L95	0.9798

Table 1: Results of applying nDCG to orderings provided by the *turkers*.

The results of applying nDCG are higher than the ones gotten from the pairwise relationship approach. Although the nDCG score is a useful metric for evaluating relevance ranking, it might not be the most appropriate metric for evaluating the results of the task performed with the *turkers* since it penalizes top ranked results more and we would like to penalize misplaced assigned summary grades according to their distance from the target reading level.

4) **The usability of summaries generated at different reading levels for users with different reading skills**. For this evaluation 16 students at the 5^{th} grade and 34 freshmen college students were recruited. They received two summaries for each of nine different graphs: one at the 4^{th} – 5^{th} and the other at the 11^{th} - college reading level. They were asked to choose which summary they preferred and why. Results per grade and per graph are presented

in Table 2 for the 5th graders and in Table 3 for college students. Additionally, they were asked to circle things they did not like in either summary. From 73 responses collected from the 5th graders, 57 chose the summaries at their reading level and from the 163 responses collected from the college students, 115 chose the summaries at their reading level. Table 4 shows that the results are statistically significant given p = 3.67816E-12 calculated using the chi-squared test.

Line graph	Chose 4th-5th	Chose 11th - cc
L6	9	3
L17	10	1
L18	1	0
L21	6	3
L26	5	2
L28	8	2
L42	7	0
L89	7	1
L95	4	4
Total	**57**	**16**

Table 2: Results from reading level experiment with 5th graders.

Line graph	Chose 4th-5th	Chose 11th - cc
L6	5	13
L17	6	14
L18	4	15
L21	6	16
L26	5	14
L28	5	15
L42	5	10
L89	6	10
L95	6	8
Total	**48**	**115**

Table 3: Results from reading level experiment with freshmen College students.

	5th graders	College students	Total	Prob
5th grader text	57	48	105	0.44
College text	16	115	131	0.56
Total	73	163	236	

Table 4: Statistical significance data.

From these results we conclude that the system is able to successfully generate summaries that match the reading level of the articles on which the line graphs appear and that its perception of text complexity matches that of human readers at a rate of 72%. In order to assess how good this result is, another possible experiment could contain the same tasks, but compare the results of our system with those obtained from a baseline. Such baseline currently does not exist. One possibility could be to provide them with summaries generated using Benetech (Benetech, 2016) guidelines for line graph description as they are made available, for example. We also confirmed our initial contention that readers with different reading abilities prefer text that matches their reading skills, instead of always reading the simplest text they can get.

8 Conclusion

This work presents novel approaches applied to the microplanning phase to enable NLG system to tailor the generated text to match different target reading levels. After identifying through an experiment that more sophisticated readers prefer more sophisticated text and that readers at lower reading levels would prefer text that was simpler, we developed and successfully evaluated a system that uses learning, a graph search algorithm with the help of a heuristic for aggregation and a lexicalization phase that chooses domain relevant and grade level appropriate lexical items when generating summaries of line graphs. This contributes to the NLG research area by describing and evaluating automated aggregation and lexicalization approaches that consider different reading abilities.

9 Future Work

For the microplanning phase of the system we envision future work on the pronominalization phase and coordination of lexical items. For the latter, we want to enable to use of different lexical items to describe the same concept in the summary by using a different referring expression. Additionally, we want to enable the system to coordinate contrasting concepts when choosing lexical items. One example is to coordinate *top* vs *bottom*, *maximum* vs *minimum*, *first* vs *last*, *higher* vs *lower*, instead of randomly selecting lexical items.

References

Barzilay, R., & Lapata, M. (2006). *Aggregation via set partitioning for natural language generation.* Paper presented at the the Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics.

Bateman, J. A., & Paris, C. L. (1989). *Phrasing a Text in Terms the User Can Understand.* Paper presented at the Proceedings of the 11th International Joint Conference on Artificial Intelligence - Volume 2, San Francisco, CA, USA.

Bayyarapu, H. S. (2011). *Efficient algorithm for Context Sensitive Aggregation in Natural Language generation.* Paper presented at the RANLP.

Benetech. (2016). Benetech.

Carroll, J., Minnen, G., Pearce, D., Canning, Y., Devlin, S., & Tait, J. (1999). *Simplifying Text for Language-Impaired Readers.* Paper presented at the In Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics (EACL.

Common Core State Standards Initiative, C. (2010). Common Core State Standards for English language arts and literacy in history/social studies, science and technical subjects. Retrieved from http://www.corestandards.org/

Covington, M., He, C., Brown, C., Naci, L., & Brown, J. (2006). *How Complex is that Sentence? A Proposed Revision of the Rosenberg and Abbeduto D-Level Scale.* Paper presented at the Research Report, Artificial Intelligence Center, University of Georgia.

Demir, S., Carberry, S., & McCoy, K. F. (2012). Summarizing Information Graphics Textually. *Computational Linguistics, 38*(3), 527-574.

Demsar, J., Curk, T. v., Erjavec, A. v., Gorup, v. r., Ho\vc, e. T. v., Milutinovi\vc, M., . . . Zupan, B. v. (2013). Orange: Data Mining Toolbox in Python. *J. Mach. Learn. Res., 14*(1), 2349-2353.

Dictionary.com, L. L. C. (2015). Thesaurus.com.

Elhadad, M., Robin, J., & McKeown, K. (1997). Floating Constraints in Lexical Choice. *Comput. Linguist., 23*(2), 195-239.

Fellbaum, C. (1998). *WordNet: An electronic Lexical Database*: The MIT Press.

FSF. (2005). Style and Diction GNU project. Retrieved from www.gnu.org/software/diction

Greenbacker, C., Carberry, S., & McCoy, K. (2011, July). *A Corpus of Human-written Summaries of Line Graphs.* Paper presented at the Proceedings of the UCNLG+Eval: Language Generation and Evaluation Workshop, Edinburgh, Scotland.

Kipper, K., Dang, H. T., & Palmer, M. (2000). *Class-Based Construction of a Verb Lexicon.* Paper presented at the Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence.

Lexile Framework for Reading. (2015).

Library, A. (2015, August). Austin Public Library Electronic Catalog. Retrieved from http://library.austintexas.gov/

Loper, E., & Bird, S. (2002). *NLTK: The Natural Language Toolkit.* Paper presented at the Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1, Stroudsburg, PA, USA.

Michel, J. B., Shen, Y. K., Aiden, A. P., Veres, A., Gray, M. K., Pickett, J. P., . . . Aiden, E. L. (2011). Quantitative analysis of culture using millions of digitized books. *Science, 331*(6014), 176-182.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *CoRR, abs/1301.3781*.

Moraes, P., McCoy, K., & Carberry, S. (2014). *Adapting Graph Summaries to the Users'' Reading Levels.* Paper presented at the Proceedings of the 8th International Natural Language Generation Conference.

Moraes, P. S., Carberry, S., & McCoy, K. (2013). *Providing access to the high-level content of line graphs from online popular media.* Paper presented at the Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility, Rio de Janeiro, Brazil.

Rello, L., Baeza-Yates, R., Bott, S., & Saggion, H. (2013). *Simplify or Help?: Text Simplification Strategies for People with Dyslexia.* Paper presented at the Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility, New York, NY, USA.

Siddharthan, A. (2003). *Preserving Discourse Structure when Simplifying Text.* Paper presented at the In Proceedings of the 2003 European Natural Language Generation Workshop.

Vajjala, S., & Meurers, D. (2012). *On Improving the Accuracy of Readability Classification Using Insights from Second Language Acquisition.* Paper presented at the Proceedings of the Seventh Workshop on Building Educational Applications Using NLP, Stroudsburg, PA, USA.

Walker, M. A., Rambow, O., & Rogati, M. (2001). *SPoT: a trainable sentence planner.* Paper presented at the Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies, Stroudsburg, PA, USA.

Wikipedia. (2004). Wikipedia, The Free Encyclopedia.

Wilkinson, J. (1995). *Aggregation in Natural Language Generation: Another Look*. Retrieved from

Williams, S., & Reiter, E. (2004, August). *Reading errors made by skilled and unskilled readers: evaluating a system that generates reports for people with poor literacy.* Paper presented at the Fourteenth Annual Meeting of the Society for Text and Discourse, Chicago.

Williams, S., & Reiter, E. (2005a). *Appropriate Microplanning Choices for Low-Skilled Readers.* Paper presented at the IJCAI.

Williams, S., & Reiter, E. (2005b). *Generating readable texts for readers with low basic skills.* Paper presented at the Proceedings of the 10th European Workshop on Natural Language Generation (EWNLG 2005).

Williams, S., Reiter, E., & Osman, L. (2003, August). *Experiments with discourse-level choices and readability.* Paper presented at the Proceedings of the 9th European Workshop on Natural Language Generation (ENLG-2003), Budapest.