Predicting Cognitive Impairments with a Mobile Application

Elif Eyigöz, Guillermo Cecchi and Ravi Tejwani
IBM Research, Yorktown Heights, NY, 10598, USA
{*ekeyigoz,gcecchi,rtejwan*}*@us.ibm.com*

Keywords: Automatic prediction of MMSE, Syntactic complexity, Cognitive impairments

Abstract: Assessment of cognitive impairments is of social and clinical importance for vulnerable populations, such as elderly, athletes and soldiers, who are prone to falling victim to cognitive impairments. This paper presents ongoing work for developing an application that predicts the neurological state of users with the state-of-the art performance through analyzing the structural complexity of users' utterances. We present a novel method that estimates the neurological state of users with Pearson correlation of 0.66 with respect to the Mini-mental state exam score. Unlike previous work, our method does not depend on assumptions of relating linguistics representations to human language-processing capabilities, but discovers the discriminative patterns automatically.

1 Introduction and motivation

In this paper, we present ongoing work on development of a mobile application that estimates the degree of cognitive impairment of a user with state-of-the-art performance, upon collecting a speech sample by prompting the user with a picture description task. We expect a large portion of our users to be people with cognitive impairments due to aging related neurodegenerative disorders, and people with traumatic brain injury.

Dementia is a growing social and clinical problem, as three percent of people between the ages of 65 and 74, 19% between 75 and 84, and nearly half of those over 85 have the condition (Umphred, 2007). Early detection of the disorder, coupled with access to care planning leads to better outcomes for both patients and their caregivers (Bradford et al., 2009). The true prevalence of missed and delayed diagnoses of dementia is unknown but seems to be very high (Bradford et al., 2009). Diagnosis of dementia is prone to be delayed, because it is dependent on suspicion and concern based on patients symptoms. A major factor for delayed diagnosis is lack of access to affordable healthcare, as patients in lower strata tend to go undiagnosed at a higher rate (Maestre, 2012). Accordingly, economic issues are also critical for control and management of dementia after diagnosis. Therefore, cost-effective, easy-to-use and naturalistic tools for routine dementia-screening and disease-progression monitoring could provide patients and medical professionals with the opportunity to engage in efficient treatment planning.

Our tool is going to be useful for assessing not only slow-developing cognitive impairments like dementia, but also for sudden changes in cognitive capabilities, for example due to a traumatic brain injury (TBI), or a stroke. In 2013, about 2.8 million TBI-related emergency department visits, hospitalizations, and deaths occurred in the United States (Sosin et al., 1996). Members of certain professions, such as athletes and combat soldiers, are more prone to falling victim to TBI (Cole et al., 2017). Currently, there are several computerized neurocognitive assessment tests used for TBI that engage various cognitive domains, such as memory, attention, motor speed, processing speed etc. (Cole et al., 2017). However, none of these tests perform language analysis using NLP technology with linguistic sophistication that can quantify structural complexity of a speakers utterances. Therefore, our tool is going to be a significant contribution to the existing battery of computerized neurocognitive assessment tools used for TBI.

In this paper, we present a novel method for estimating the degree of cognitive impairment, and also describe our efforts on building a prototype. We validate our method by performing regression to predict the Mini-Mental State Examination (MMSE) score. MMSE is a neuropsychological test that is used extensively in clinical research to estimate the severity and progression of cognitive impairment (Folstein et al., 1975; Pangman et al., 2000). It is the best

studied and the most commonly used test for the diagnosis and longitudinal assessment of Alzheimer's Disease -the most common type of dementia (Burns and Iliffe, 2009). MMSE is also considered as an effective way to document an individual's response to treatment (Pangman et al., 2000). MMSE is also used for evaluating cognitive outcome in patients with TBI, both immediately following an injury and in the follow-up period, although its sensitivity depends on the site of injury (Lee et al., 2015; De Guise et al., 2013).

We use NLP techniques, in particular syntactic analysis of constituent parse trees for feature extraction. To validate our method, we used data from the Pitt Corpus, which is part of the publicly available DementiaBank corpus (Macwhinney et al., 2011). Our method can successfully estimate the cognitive impairment of subjects in the DementiaBank study with Pearson correlation of 0.66 with respect to MMSE, which is currently the state-of-the art in predicting MMSE.

The outline of the paper is as follows: We first summarize related work in Section 1.1; we then provide necessary background for presenting our feature-extraction method in Section 2, present the feature-extraction method for predicting MMSE in Section 3, and the feature-selection method in Section 4; we discuss the performance of our method in Section 6, and describe the current status of our implementation and future development plans in Section 7.

1.1 Related work

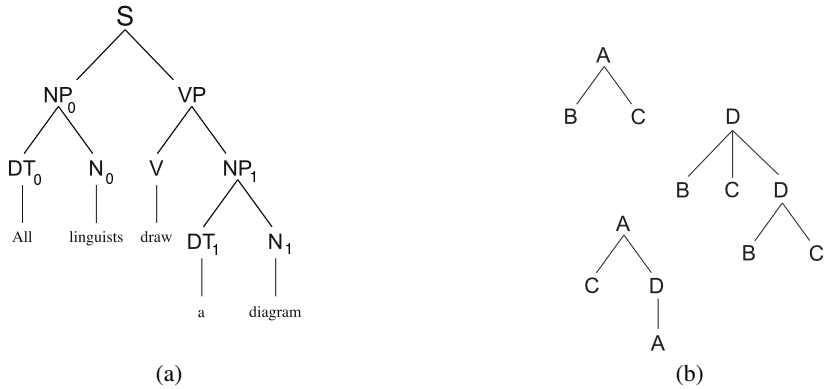
There is growing number of papers in the recent years using the DementiaBank corpus, predominantly doing classification of patients vs health controls (Fraser and Hirst, 2016; Fraser et al., 2016; Orimaye et al., 2014, 2017). Fraser et al. (2016) use syntactic, semantic, lexical and acoustic features to classify Alzheimer's disease patients vs healthy controls in DementiaBank. They used context free grammar (CFG) rule rates and proportions in sample interviews as features, in addition to average length of the right hand side of CFG rules, only for noun phrases (NP), verb phrases (VP), and prepositional phrases (PP). They obtained 81 percent classification accuracy, however the results they reported were obtained with features that were selected using the entire data set. Orimaye et al. (2014) and Orimaye et al. (2017) also used syntactic features for classification of Alzheimer's disease patients vs healthy controls in DementiaBank. They used syntactic features involving sentence embeddedness, in particular they focused on POS tags indicating coordinated, subordinated, and reduced sentences (CC, S, VBG, VBN). They also used counts of unique CFG rules, the valency of verbs, lexical features involving repetition. Their results were also obtained using features that were selected using the entire data set.

To the best of our knowledge, the only study that predicted MMSE scores using linguistic features is Yancheva et al. (2015), where they modeled longitudinal progression of MMSE scores using subjects that have more than one sample in DementiaBank. They reported a mean-absolute-error (MAE) of 2.91 in predicting MMSE, significantly below the within-subject inter-rater standard deviation of 3.9 to 4.8 (Molloy et al., 1991). However, the lowest MAE they obtained with a method generalizable to unseen data was 7.31, as they also reported results obtained with using features that were selected using the entire data set. They did not report the correlation between the scores their method predicted and the actual MMSE scores.

Our work differs from most psycholinguistics and neuroscience studies on linguistic aspects of neurological disorders in multiple ways: First and foremost, our method is not intended for theoretical understanding of human language production and processing capabilities, but for practical applications.

Second, we present results that are generalizable to unseen data. We perform feature-selection in each cross-validation (CV) fold separately without observing the entire data set, and use all features selected in the folds of CV, as opposed to related work that report results obtained with features selected using the entire data set.

Finally, a major difference between related work and our method is that out method does not depend on assumptions relating linguistics representations to human-language-processing capabilities. Prior studies all use hand written rules involving node labels (e.g. NP, VP, S), naturally supported by the psychologists literature, for feature extraction. Our method, on the other hand, does not depend on the actual syntax-tree labels. For example, the tree in Figure 1(a) has node labels that are commonly used in language studies, however the trees in Figure 1(b) have node labels that are variables. Our method can be used on either types of trees, and thus can be applied to trees of any syntactic theory, as long as we can obtain parse trees of utterances for training and testing. A major advantage of this approach is that we can, and do, apply our method to languages other than English, which do not use familiar node labels.



(a) (b)

Figure 1: Example syntax trees.

2 Background

2.1 Syntax trees

We model the structural complexity of users' utterances through syntactic analysis. In particular, we analyze constituency parse of the utterances, which are obtained by a statistical parser. In this section, we define basic syntactic tree relations in order to provide a background for our feature extraction method.

Figure 1(a) shows a constituency parse for the sentence "All linguists draw a diagram". As shown in Figure 1(a), parse trees are graphs that are made up of nodes (e.g. NP, VP), and edges that connect the nodes. Nodes have labels, such as NP for a noun phrase, VP for a verb phrase etc. The node labels in this study follow the convention of constituent tags in Penn Treebank (Marcus et al., 1993), as we used a statistical parser trained on Penn Treebank to parse our data. However, our method does not depend on having prior knowledge of actual node labels, as we could have used some other method, and obtained trees with a different label set, for example if we apply our method to another language.

Nodes NP_0 and NP_1 have the same label, NP, as the subscripts are not parts of the node labels. We use the subscripts, e.g. NP_0, on the labels in Figure 1(a) for disambiguation between the nodes with the same label. Branch out of a tree node is conveyed in a context free grammar (CFG) rule, for example as 'NP $\rightarrow$ DT N', which is instantiated twice in Figure 1(a): NP_0 that branches to DT_0 and N_0, and NP_1 that branches to DT_1 and N_1. A CFG rule covers all and only the nodes that branch out of a single node. Our work differs from related work using syntax, in that they mostly use CFG rules, but not smaller or larger subtree patterns.

In the definitions of this paper, we assume trees have only directed edges, in that the nodes can be traversed through the edges only in a top down fashion. For example, there exists a path to all nodes in the tree from node S, however there is no path that can reach node S, as it is at the top of the tree and nodes can be traversed only in a top down fashion. For this reason, node S is called the root of the tree, and syntax trees have a single root node. The tree under the root node covers the entire sentence, whereas trees under other nodes span subparts of the sentence. For example, the tree under the node with label VP spans "draw a diagram". The subspan covered by a node is called the *yield* of that node. In Figure 1(a), there are two NPs, one has the yield "all linguists", and the other has the yield "a diagram". Some nodes have yields that are only single words. For example, the node with label N_1 yields only the word "diagram".

2.2 Relations between nodes

In this section, we define tree relations that span smaller and larger subtrees than CFG rules. The most primitive relations between nodes in a tree are *mother*, *sister*, *dominance* relations. Node *a* dominates node *b*, if and only if (iff) there is a path between the root node and *b* which passes through *a*. For example, NP_0 dominates DT_0, and VP dominates N_1. Node *a* is the mother of node *b*, iff *a* dominates *b*, and there is an edge between *a* and *b*. For example, node NP_1 is the mother of node DT_1. Node *a* is a sister of node *b*, iff they have the same mother. For example, node DT_1 and node N_1 are sisters.

The root node S in Figure 1(a) dominates DT_1, and the path between them has three edges. Therefore, the dominance relation can have a larger scope than CFG rules, which can only represent relations of depth one. For feature extraction, we use dominance relations between nodes that are two edges apart.

Table 1: Example counts of subtree patterns.

Unary			Binary			Ternary		
Node-label	Count	Rate	Sister	Count	Rate	C-command-via-node	Count	Rate
A	3	3/13	Sister(B,C)	3	3/6	Comm-max(B,B,D)	1	1/5
B	3	3/13	Sister(C,D)	2	2/6	Comm-max(B,C,D)	1	1/5
C	4	4/13	Sister(B,D)	1	1/6	Comm-max(C,B,D)	1	1/5
D	3	3/13				Comm-max(C,C,D)	1	1/5
						Comm-max(C,A,A)	1	1/5
Node total	13		Sister total	6		Comm-max total	5	

In addition, we make use of tree relations with larger scope than CFG rules that were first defined within the Chomskyan tradition. The first rule is *c-command* (Haegeman, 1994). Within the scope of this paper, c-command between two nodes could be described informally as an *aunt* relation, if we continue with the analogy of sisterhood and motherhood as relations between nodes. Formally, c-command is defined as follows: node *a* c-commands node *b*, iff *a* does not dominate *b*, *b* does not dominate *a*, and the lowest branching node *c* that dominates *a* dominates *b*. For example in Figure 1(a), node V c-commands N_1, -i.e. V is the aunt of N_1- because V does not dominate N_1, N_1 does not dominate V, and VP is the mother of V, which dominates N_1. The formally-defined c-command relation can span nodes of arbitrary distance. However, we limit our feature-extraction method to consider only the most local c-command relations, which can informally be defined as *aunt* relations. According to the formal definition, sister relation is also a c-command-relation. However, we do not consider sister relations as c-command relations in this paper, as we already use sister relations for feature extraction, and want to keep c-command and sister relations mutually exclusive for feature extraction.

Next, we define a ternary version of the c-command relation: c-command is defined between two nodes, *a* and *b*, whereas *c-command-via-node* is defined not only between *a* and *b*, but also includes *c*, where node *c* could be defined *informally* as the grandmother of node *b*. Formally, node *a* c-commands *b* via *c*, iff *a* does not dominate *b*, *b* does not dominate *a*, and the lowest branching node *c* that dominates *a* dominates *b*. For example in Figure 1(a), node V c-commands N_1 via VP. Within the scope of this paper, this relation is constrained to cover only ternary relations that exists between a node, the nodes aunt, and the nodes grandmother, as defined informally.

Finally, we also use a ternary *dominate-via-node* relation, that includes a node, the nodes mother, and the nodes grandmother, where a node is dominated by its grandmother via its mother. To summarize, we use the following relations for feature extraction: sister, dominate, c-command, c-command-via-node, dominate-via-node, and the node labels.

3 Feature extraction

3.1 Subtree patterns

We illustrate our method of feature extraction with the example trees in Figure 1(b). There are multiple trees in Figure 1(b), as most samples consist of multiple utterances. In the example, the node labels are not from the Penn-Treebank constituent tag set. Instead, we used variables for node labels in the example, first to emphasize that our method does not depend on the actual node labels, but their relations in the trees, as defined in Section 2. Second, because it is common to use variables for node labels for generalizability, and also for easy-readability.

Table 1 lists instances of node labels, sister relations, and c-command-via-node relations observed in the trees in Figure 1(b), and their counts. Let us first look at the examples of node labels in the first column in Table 1. Total number of nodes in the trees in Figure 1(b) is given in the last row. We divide the count of a node label by the total count of nodes. For example in Figure 1(b), there are three nodes labeled as B, and the total count of nodes is 13. Thus the rate of nodes labeled B is 3/13.

In the second column in Table 1, we show counts of sister relations in the trees in Figure 1(b). We observe node B and C as sisters three times, C and D as sisters two times, B and D as sisters only once. For each sister relation instance, we divide the count of

that instance by the total number of sister relations. For example in Figure 1(b), the count of sister(B,C) is three, the total count of sister relations is six. Thus, the rate of sister(B,C) is 3/6. The counts of all sister relation instances, and the total count of sister relations are shown in Table 1. In sum, a rate is obtained for each instance of the sister relation by dividing the count of that instance by the sum of the counts of all instances of the sister relation.

Similarly, the rates of instances of the c-command-via-node relation are computed in the same manner, as shown in Table 1. For all relations mentioned in the previous section, we obtain a rate for each instance of the relation by dividing the count of that instance by the sum of the counts of all instances of that relation. We use logarithm of the rates as features, and use 10e-7 as floor in order to avoid computing the logarithm of zero.

Finally, we also use features involving CFG rules: We normalize the counts of instances of CFG rules by the total number of CFG rules in a sample. For example, the CFG rule A $\rightarrow$B C occurs only once in Figure 1(b), and there are total five CFG rules in Figure 1(b), thus the rate of A $\rightarrow$B C is 1/5. We also use statistics over the length of CFG rules -as the number of nodes at the right side of CFG rules- in a sample. We compute minimum, maximum, mean, standard deviation and percentiles over the length of CFG rules as features.

3.2 Node scores

Statistical parsing algorithms compute a score between 0 and 1 for each node, indicating how grammatical the yield of a node is within the context of the entire sentence. We obtain the node scores from the statistical parsers data structures. For each node label, we compute statistics over the scores assigned to the nodes with that label in the sample. We compute maximum, minimum, standard deviation, skewness and kurtosis over the node scores for each label, and use them as features.

4 Feature-selection methods

As we do not assume prior knowledge of what node labels or subtree patterns indicate in terms of syntactic complexity with respect to human language processing, we generate a large number of features for all observed subtree patterns in the samples. As a res==ult, eliminating features of low quality is essential to the performance of our method. We resorted to an experimental method for eliminating low quality features within a leave-one-subject-out cross-validation setting (LOOCV). We split the data to folds of train-test sets. Within each fold, we performed feature-selection as explained below, and predicted the scores of the samples from the left-out-subject using the selected features.

Univariate feature selection As initial filtering, we used univariate feature selection methods. We obtained a p-value for each feature by computing Pearson r between the feature and the MMSE scores. We eliminated features with p-value greater than 0.01. Then, we performed an ANOVA-F test, and modeled the decreasing p-values as an exponential-decay curve. We used curve fitting to obtain the τ parameter for the decay curve. We learned a multiplier a for the τ parameter with cross-validation, where $\alpha \cdot \tau$ is used as a threshold to eliminate features that are at the tail of the exponential-decay curve.

Stability-selection We followed univariate selection methods with stability-selection (Meinshausen and Bühlmann, 2010). We used the `scikit-learn` implementation of Randomized Lasso, which returns a score for each feature. We modeled the decreasing feature scores as an exponential-decay curve. We used curve fitting to obtain the τ parameter, and learned a multiplier a for the τ parameter with cross-validation, where $\alpha \cdot \tau$ is used as a threshold to eliminate features that are at the tail of the exponential-decay curve.

Recursive-feature-elimination Next, we used recursive feature elimination (RFE). Given an estimator, RFE selects features by recursively considering smaller and smaller sets of features. First, the estimator is trained on the initial set of features and weights are assigned to each one of them. Then, features whose absolute weights are smallest are pruned from the current set features. This procedure is recursively performed on the pruned set until the features are exhausted.

`RFECV` in the `scikit-learn` package performs RFE in a cross-validation loop to find the optimal set of features. `RFECV` requires an estimator to obtain weights for the features, for which we used Linear Regression. As `RFECV` returns an optimal set of features, we rerun `RFECV` using the optimal set returned by the previous run, until it no longer returned a smaller set. In other words, we repeated `RFECV` until it converged.

Feature-selection in LOOCV Within each fold of LOOCV, we started with univariate feature-selection

methods, as they can quickly eliminate a large number of features of low statistical significance. Eliminating large number of features is critical for subsequent feature selection methods, namely stability-selection and recursive-feature-elimination, as they can be a lot slower than the univariate feature-selection methods.

Recursive-feature-elimination, unlike stability-selection, can be unstable across folds in terms of the number of features it eliminates. For that reason, we perform stability-selection before recursive-feature-elimination. Performing recursive-feature-elimination as the last step of feature-selection ensures that it evaluates only a small number of features in each fold, thus the instability of the method can be relatively constrained.

Finally, the selected features in each fold were used for training, and the scores of the test samples were predicted using the fitted estimators.

5 Data

We validated the feature extraction method explained in the previous section for prediction of MMSE using the publicly available DementiaBank corpus (Macwhinney et al., 2011). Patients of various types of dementia were included in the study, in addition to age and education matched healthy controls. Demographics of DementiaBank can be found in Table 2.

All subjects were given the Cookie Theft picture description task from the Boston Diagnostic Aphasia Examination (Kaplan, 1983). This task was chosen, because it is considered an ecologically valid approximation to spontaneous discourse. We used each narrative for the description as a sample, and parsed the utterances using the Stanford Parser (Klein and Manning, 2003). All subjects were associated with a professionally administered MMSE score on a scale of 0 (greatest cognitive impairment) to 30 (no cognitive impairment).

Table 2: DementiaBank Demographics.

	Dementia	Control
Number of samples	278	182
Number of subjects	192	96
Age (years)	72 (8.66)	64 (7.48)
Gender (male/female)	101/177	66/115
MMSE	20 (5.7)	29 (1.1)

Table 3: Feature-selection results for the baseline model, which uses only features involving CFG rules, and the best performing model, which uses all subtree patterns. The second column shows the median number of features selected across folds by the method given in the first column. The feature-selection methods are listed sequentially with respect to their application.

(a) CFG Rules

Total # Features	978	Pearson r	MAE
1 Pearson r <0.01	65	0.60	4.08
2 ANOVA f-test	20	0.61	3.94
3 Stability	18	**0.61**	**3.93**
4 RFE	16	0.60	3.95

(b) Subtree patterns.

Total # Features	4297	Pearson r	MAE
1 Pearson r <0.01	468	0.62	3.97
2 ANOVA f-test	99	**0.66**	**3.86**
3 Stability	35	0.64	3.91
4 RFE	35	0.64	3.91

6 Experiments and results

Features involving CFG rules were commonly used in previous work for classification of patients vs controls in DementiaBank. Therefore, we use features involving CFG rules as a baseline model. Please note that we used *all* CFG rules observed in the interviews, not only CFG rules involving a pre-determined set of node labels. We have in total three experimental conditions:

- Baseline: CFG features
- All subtree patterns: CFG features plus features involving other subtree patterns, e.g. sister, dominance, and c-command relations, as explained in Section 3.1.
- Node scores: As explained in Section 3.2.

Table 3 shows the results obtained after each feature-selection step for each experimental condition. We do not report these detailed feature-selection results for the node scores experiment, as it was the worst performing experimental condition, as shown in Table 4. It shows the sequential decrease in the number of features after each feature-selection step. In both experiment conditions, the largest decrease in the number of features was obtained by the first feature-selection step, and the smallest decrease in the number of features was obtained by the last feature-

Table 4: The best estimators.

	Subtree patterns	CFG rules	Node scores
# Features	35	16	34
Pearson *r*	0.64	0.60	0.56
Estimator	Lasso CD	All	Ridge CD
MAE	3.91	3.95	4.29
Estimator	Ridge / Elastic Net	eSVR	Linear Regression

selection step. In Table 3, we report results obtained with the selected features in terms of Pearson *r* correlation between the predicted scores, and the actual MMSE scores. In addition, we report mean-absolute-error (MAE) between the predicted scores and the actual MMSE scores. Although MAE is a scale-dependent measure, we have to report results with this metric, as previous results on predicting MMSE scores both automatically and manually have been reported in terms of MAE.

Table 4 shows the best performing estimators on the smallest number of features after all feature-selection steps had been applied. In Table 4, the first row shows the experimental conditions, the second row shows the median number of features selected across folds per experimental condition. The row for Pearson *r* shows the correlation between the predicted MMSE scores and the actual MMSE scores. The next row shows the estimator that obtained the Pearson *r* performance. The MAE row shows the mean-absolute-error between the predicted MMSE scores and the actual MMSE scores. The row under the MAE row shows the estimators that obtained the MAE performance. The complete list of estimators we experimented with, along with their initialization and grid search parameters can be found in the Appendix. In Table 4, "All" stands for all estimators given in the Appendix.

6.1 Discussion

The highest correlation for prediction of MMSE was obtained by using all subtree patterns, with Pearson *r* of 0.66, as shown in Table 3(b). The improvement in Pearson *r* over the baseline, as shown in in Table 3(a), was five percent. Therefore, using subtree patterns that have smaller scope than CFG rules, such as sister relations, and subtree patterns that have larger scope than CFG rules, such as c-command relations, improved performance.

Table 3(b) shows that, the performance was optimal after using only the first two feature-selection steps: using a p-value threshold and the ANOVA f-test. However, stability-selection resulted in a large drop in the number of features with a minor decrease in performance. On the other hand, RFE resulted in a minor decrease in the number of features, and performance, in both experiment conditions. Following the Occam's razor principle, we decided to use the settings with the smallest number of features, and the estimators that performed best with the smallest number of features, as seen in the second column of Table 4, in our application, despite the minor decrease in performance on our data.

Using only node scores provides 0.56 Pearson *r* correlation, which shows that node scores, computed by a statistical parser solely for algorithmic purposes can convey information with cognitive significance. These results are in line with the interpretation of node scores as indicating grammaticality of constituents. However, we observed that combining node scores with subtree patterns did not improve performance over using only subtree patterns.

Our best mean-absolute-error (MAE) score is 3.86, which is comparable to within-subject inter-rater standard deviation of 3.9 to 4.8 (Molloy et al., 1991). Yancheva et al. (2015) reported a MAE of 2.91, however the lowest MAE they obtained with a method generalizable to unseen data is 7.31. They used the *entire* data set to learn a hyper-parameter: the optimal feature-set size for best performance on the *entire* data set. Thus, although they used leave-one-subject-out cross-validation, the test-sets in their LOOCV folds effectively became *validation* sets for learning this hyper-parameter. As a result, their results are not generalizable to unseen data. On the other hand, we performed feature-selection within the training set of each fold, not using the samples in the test-sets of LOOCV.

6.1.1 Selected-features

An examination of the features that have survived the feature-selection process in each fold shows that our method made use of features that have commonly been suggested as relating to human-language-

processing capabilities, and have been used in prior work. These features fall in four categories:

- Subtree patterns involving predicate argument structure. For example, sister relations involving modifiers, e.g. adjectives and adverbs.
- Subtree patterns involving sentence embeddedness.
- Ungrammatical parses, due to disfluencies. For example, double determiners for "the the".
- Statistics over CFG rule length.

Our method had the advantage that we did not have to hand-code rules involving the node labels, but rather use machine-learning techniques discover the features among thousands of features generated using only a few subtree patterns.

We have also observed that patterns that have smaller scope than CFG rules are more useful than patterns that have larger scope than CFG rules. It seems that factoring out CFG rules into even smaller tree relations allows us to extract more fine grained features, which in turn improve learning performance.

7 Prototype

Initial deployment of the mobile application will be for end-users that are the diagnosed patients of aging-related neurodegenerative disorders enrolled in a clinical study aimed at assessing drug effectiveness. The scores predicted by the system will be provided to medical professionals for evaluation.

Upon authentication, the app will prompt the user with a picture description task, and request the user to complete a short questionnaire. The questionnaire will include a few questions to control for confounding factors such as general status of health, stress level, alcohol consumption etc., to be used for elimination of samples that were recorded under unfavorable conditions.

The initial release of the tool to the medical professionals will include the cookie-theft description task, not only because we validated our method on this task, but also it is the most-commonly used task for eliciting syntactically complex utterances (Spreen and Risser, 2003). Further deployments will include other picture description tasks that have been accepted and used by the scientific community as valid tasks for eliciting syntactically complex utterances.

The tool will be deployed on IBM Bluemix platform, which offers the following services: speech-to-text for transcribing speech samples to text, NLP analysis tools for obtaining parse trees and node scores from transcribed text, HIPAA (U.S. Department of Health and Human Services, 2003) compliant and scalable data services. We have already developed a test system that has speech-to-text and NLP analysis capabilities. We plan to deploy a test system for internal-use for the whole pipeline described in this paper within a year.

8 Conclusion

We reported ongoing work on developing a tool that estimates the degree of cognitive impairment of a user with state-of-the-art performance comparable to human inter annotator reliability scores. We presented a novel feature extraction method for prediction of MMSE, and also a feature-selection methodology that discovers useful features in a way that is generalizable to unseen data.

A major advantage of our method over prior work is that it does not rely on human determined set of syntactic patterns, but discovers the discriminative patterns automatically among all observed syntactic patterns. As a result, it can be applied to trees generated under different syntactic assumptions, such as trees of different languages, without any supervision from linguists or subject-matter experts.

We estimate and hope that our mobile application will have wide practical applicability in both clinical and in-home use.

REFERENCES

Bradford, A., Kunik, M. E., Schulz, P., Williams, S. P., and Singh, H. (2009). Missed and delayed diagnosis of dementia in primary care: prevalence and contributing factors. *Alzheimer Disease & Associated Disorders*, 23(4):306–314.

Burns, A. and Iliffe, S. (2009). Dementia. *BMJ*, 338:b75.

Cole, W. R., Arrieux, J. P., Ivins, B. J., Schwab, K. A., and Qashu, F. M. (2017). A Comparison of Four Computerized Neurocognitive Assessment Tools to a Traditional Neuropsychological Test Battery in Service Members with and without Mild Traumatic Brain Injury. *Archives of Clinical Neuropsychology*, pages 1–18.

De Guise, E., Leblanc, J., Champoux, M. C., Couturier, C., Alturki, A. Y., Lamoureux, J., Desjardins, M., Marcoux, J., Maleki, M., and Feyz, M. (2013). The mini-mental state examination and the Montreal Cognitive Assessment after

traumatic brain injury: an early predictive study. *Brain Injury*, 27(12):1428–1434.

Folstein, M., Folstein, S., and McHugh, P. (1975). "mini-mental state". a practical method for grading the cognitive state of patients for the clinician. *Journal of Psychiatric Research*, 12(3):189–198.

Fraser, K. C. and Hirst, G. (2016). Detecting semantic changes in alzheimers disease with vector space models. In *Proceedings of LREC 2016 Workshop. Resources and Processing of Linguistic and Extra-Linguistic Data from People with Various Forms of Cognitive/Psychiatric Impairments (RaPID-2016)*, number 128. Linköping University Electronic Press.

Fraser, K. C., Meltzer, J. A., and Rudzicz, F. (2016). Linguistic features identify Alzheimers disease in narrative speech. *Journal of Alzheimer's Disease*, 49(2):407–422.

Haegeman, L. (1994). *Introduction to Government and Binding Theory*. Blackwell Textbooks in Linguistics. Wiley.

Kaplan, E. (1983). *The assessment of aphasia and related disorders*, volume 2. Lippincott Williams & Wilkins.

Klein, D. and Manning, C. D. (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 423–430. Association for Computational Linguistics.

Lee, C. N., Koh, Y. C., Moon, C. T., Park, D. S., and Song, S. W. (2015). Serial Mini-Mental Status Examination to Evaluate Cognitive Outcome in Patients with Traumatic Brain Injury. *Korean J Neurotrauma*, 11(1):6–10.

Macwhinney, B., Fromm, D., Forbes, M., and Holland, A. (2011). AphasiaBank: Methods for Studying Discourse. *Aphasiology*, 25(11):1286–1307.

Maestre, G. E. (2012). Assessing dementia in resource-poor regions. *Current Neurology and Neuroscience Reports*, 12(5):511–519.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.

Meinshausen, N. and Bühlmann, P. (2010). Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473.

Molloy, D. W., Alemayehu, E., and Roberts, R. (1991). Reliability of a Standardized Mini-Mental State Examination compared with the traditional Mini-Mental State Examination. *The American Journal of Psychiatry*, 148(1):102–105.

Orimaye, S. O., Wong, J. S., Golden, K. J., Wong, C. P., and Soyiri, I. N. (2017). Predicting probable alzheimers disease using linguistic deficits and biomarkers. *BMC Bioinformatics*, 18(1):34.

Orimaye, S. O., Wong, J. S.-M., and Golden, K. J. (2014). Learning predictive linguistic features for alzheimers disease and related dementias using verbal utterances. In *Proceedings of the 1st Workshop on Computational Linguistics and Clinical Psychology (CLPsych)*, pages 78–87. sn.

Pangman, V. C., Sloan, J., and Guse, L. (2000). An examination of psychometric properties of the mini-mental state examination and the standardized mini-mental state examination: implications for clinical practice. *Applied Nursing Research*, 13(4):209–213.

Sosin, D. M., Sniezek, J. E., and Thurman, D. J. (1996). Incidence of mild and moderate brain injury in the United States, 1991. *Brain Injury*, 10(1):47–54.

Spreen, O. and Risser, A. H. (2003). *Assessment of aphasia*. Oxford University Press.

Umphred, D. (2007). *Neurological Rehabilitation*. Neurological Rehabilitation (Umphred) Series. Mosby Elsevier.

U.S. Department of Health and Human Services (2003). HIPAA privacy rule and public health. Guidance from CDC and the U.S. Department of Health and Human Services. *MMWR Supplements*, 52:1–17.

Yancheva, M., Fraser, K., and Rudzicz, F. (2015). Using linguistic features longitudinally to predict clinical scores for alzheimers disease and related dementias. In *6th Workshop on Speech and Language Processing for Assistive Technologies (SLPAT)*, pages 134–139. sn.

APPENDIX

Initialization parameters of the estimators:

```
from sklearn.linear_model import
    ElasticNet, Lasso, Ridge,
    LinearRegression
```

```
from lightning.regression import
   CDRegressor, LinearSVR
from sklearn.svm import SVR,NuSVR
 "LinearRegression":
    LinearRegression()
 "Elastic_Net": ElasticNet(max_iter
    =int(1e3))
 "Ridge_CD": CDRegressor(max_iter
    =200, tol=1e-3, loss='squared',
     penalty='l2')
 "Lasso_CD": CDRegressor(max_iter
    =200, tol=1e-3, loss='squared',
     penalty='l1', debiasing= True)
 "Lasso": Lasso()
 "Ridge": Ridge()
 "eSVR": SVR(kernel='linear')
 "NuSVR": NuSVR(kernel='linear')
 "lightSVR": LinearSVR()
```

Grid search parameters of the estimators:

```
 "LinearRegression":{"normalize":[
    False,True], "fit_intercept":[
    True,False]}
 "Elastic_Net": {"alpha": np.
    logspace(-2, 4, 5), "l1_ratio":
     10**np.array([-3,-2, -1, np.
    log10(.5), np.log10(.9)])}
 "Ridge_CD": {"alpha": np.logspace
    (-2, 2, 5)}
 "Lasso_CD": {"alpha": np.logspace
    (-2, 2, 5)}
 "Lasso": {"alpha": np.logspace(-2,
     2, 5)}
 "Ridge": {"alpha": np.logspace(-2,
     2, 5)}
 "eSVR": {"C": np.array([1, .1,
    .01, .001]), "epsilon": np.
    array([.1, 1, 5, 10, 20])}
 "NuSVR": {"C": np.array([1, .1,
    .01, .001]),"nu": np.array([.1,
     .3, .5])}
 "lightSVR": {"C": np.array([1, .1,
     .01, .001]), "epsilon": np.
    array([.1, 1, 5, 10, 20])}
```