Characterizing Facial Expressions By Grammars of Action Unit Sequences – A First Investigation Using ABL

Michael Siebers[a,*], Ute Schmid[a], Dominik Seuß[a,1], Miriam Kunz[b], Stefan Lautenbacher[b]

[a] *Cognitive Systems Group, Faculty Information Systems and Applied Computer Science, Otto-Friedrich-Universität Bamberg, 96045 Bamberg, Germany*
[b] *Institute of Psychology, Faculty Humanities, Otto-Friedrich-Universität Bamberg, 96045 Bamberg, Germany*

Abstract

We investigate the application of grammar inference to the analysis of facial expressions to discover underlying sequential regularities characteristic for a specific mental state. The input consists of sequences of action units (AUs), which represent basic facial signals. The typical classification task for facial expression analysis is to assign a *set* of AUs its corresponding mental state, e.g., an emotion. To our knowledge, there is no research investigating whether there is diagnostic information in the *sequence* in which the AUs occur in a given time interval. Our study is based on data of facial expressions of pain obtained in a psychological experiment with 347 pain episodes of 86 subjects represented as sequences of AUs. We applied the Alignment-Based Learning (ABL) approach to infer the underlying grammar for the set of all AUs which occurred in the sequences and for a reduced alphabet of the relevant AUs only. We used 10-fold cross-validation to estimate performance and we extended ABL with a frequency-based heuristics to reduce the number of grammar rules by eliminating such rules which do not contribute significantly to performance. The resulting grammar for the reduced AU alphabet provides a first approximation for a "grammar of pain".

Keywords: grammar of pain, alignment-based learning, heuristic rule extraction, facial expression analysis

1. Introduction

The most common scenario for machine learning is generalization from labeled data [3, 23]—so, samples of pre-classified data are available. In contrast, data mining research is often concerned with unlabeled data [13]—so, only datasets where it is not determined which data belong to which class or where suitable classes underlying the data are not even known are available. The majority of research addresses knowledge discovery

*Corresponding author
Email address: michael.siebers@uni-bamberg.de (Michael Siebers)
[1] Present address: Fraunhofer Institute for Integrated Circuits IIS, 91058 Erlangen, Germany

Link to published article: doi:10.1016/j.ins.2015.10.007

in *large* databases where data are represented as feature vectors and typical methods are cluster analysis or association rules [20]. In some application domains, knowledge discovery methods are applied to small sets of structured data. A classical example is Lenat's Automated Mathematician [16] where interesting relations between numbers were explored and concepts such as prime numbers were re-discovered. Another example is identifying recurring regularities in problem solving traces represented as sequences of actions [28]. Both applications aim to discover regular patterns in structured data.

A generic method to identify regular abstract patterns from smaller sets of data is grammar inference (GI). GI is a sub-field of machine learning and artificial intelligence which addresses the induction of structure from sequences of symbols [7, 11, 25, 26]. A variety of algorithms and systems learning grammars is available. The majority of work in GI is concerned with formal analyzes of learnability. However, there are several areas of application of GI, especially in the domains of natural language acquisition, computational biology, and web mining [8].

We want to explore an application of GI for structural pattern recognition in the domain of facial expression analysis [10]. More specifically, we apply GI to data of mimic reactions to systematically applied pain stimuli and identify underlying regularities in the sequences of basic facial reactions. In contrast to typical application domains of knowledge discovery, only a small set of data is available because of the high costs of data collection in this domain. Furthermore, we are interested in sequential data, i.e., a special case of structured data. Finally, the identified patterns need to be represented symbolically to be able to communicate the learned rules to experts in the application domain of pain research. To our knowledge, the applicability of GI in the domain of facial expression analysis was explored for the first time in [29]. In this paper we present an extension of this preliminary work.

In the following, we first introduce our application domain. Afterward, we describe how Alignment-Based Learning (ABL) [34] can be applied to sequences of facial expressions. The aim is to obtain compact grammars which can be interpreted by humans, especially researchers in psychology. We apply cross-validation to estimate the precision of the induced grammars. Furthermore, we present a heuristic method for reducing sets of grammar rules in such a way that loss in precision is not significant. We conclude with a short discussion and further work to be done.

2. Facial Expressions of Pain

Automated analysis of facial expressions is an active area of research since the 1990s [10, 15]. Besides applications in human-robot interaction and image retrieval, facial expression analysis is used to support the detection and analysis of human emotions and other psychological states [6]. One expression analysis approach is based on the identification of facial action units using the Facial Action Coding System [FACS, 9, 17]—an anatomically based system defined over 43 different facial movements, the so called action units (AUs). It is widely accepted in psychology that emotional states can be identified by the presence of specific sets of AUs [9]. For example, disgust is characterized by AUs 9 (*nose wrinkler*), 15 (*lip corner depressor*), and 16 (*lower lip depressor*). FACS is also used to identify pain [2, 14, 19, 27]. Typically, to identify mental activities, classifiers over AUs are learned from labeled data [10].

Σ = {au7, au4, au25, au6, au26, au9, au18, au10, au17, au14, au12, au43, au1,
au25-26, au6-7, au2, au1-2, au20, au24, au32, au16, au4-7, au5, au23,
au6-7-9, au15, au38, au19, au6-9, au4-9, au19-25-26, au7-10, au6-7-10,
au17-24, au4-7-9, au4-6-7, au30, au28, au6-12, au6-10, au7-9, au4-6-7-10,
au9-25, au4-43, au31, au45, au4-6-7-9, au39, au7-25, au34, au7-18-25,
au27, au14-24, au17-18, au15-17, au10-25, au7-20, au7-18, au4-16, au6-7-12,
au4-7-10, au6-43, au9-17, au19-25, au7-12, au12-25,au6-9-25-26, au4-7-26,
au14-16, au6-7-20, au6-20, au10-12, au6-7-25, au20-25, au4-17, au6-7-9-17}

Figure 1: List of AUs and AU compounds sorted by decreasing frequency of occurrence.

For manual as well as for automated classification the usual procedure is to identify which AUs occur in which intensities in a given time frame, e.g., 5 seconds. That is, the current practice is to identify the *set* of AUs presently active in a face to assign the underlying mental state. However, there might be additional diagnostically relevant information in the *sequence* in which the AUs appear. Knowledge about empirically valid sequences of AUs might also contribute for more realistic emotion generation in avatars and humanoid robots [21].

To explore whether AUs appear in specific sequences, we analyzed data which were obtained in a psychological study of facial expression of pain in patients with dementia by Kunz et al. [14]. The main aim of this study was to investigate whether facial responses of demented patients are identical to the responses of healthy subjects. If this were the case, facial expressions of pain could be used as an alternative pain assessment tool for demented patients, especially if the patients are verbally handicapped [18].

In the psychological study demented patients and healthy persons were exposed to pain of various intensities. Pain was induced by short pressure stimulations ranging from 1 to 5 kg, which is the physical unit of the Fischer algometer used in this study. Each subject received 4 series of 1, 2, 3, 4, and 5 kg stimuli (totaling to 20 stimulations per subject).[2] The facial responses were analyzed by a trained FACS coder and the identified AUs were entered in a database with AU number, onset-time, offset-time, and intensity rating (if appropriate). The details of the experimental setting are reported in Kunz et al. [14].

The intensity of facial responses varies widely between individuals. Even for rather strong stimuli there are typically about 1/3 non-responders, which are persons who do not show any facial reaction at all. This was also the case for the study of Kunz et al. [14]. Therefore, we only included pain episodes which occurred during high pressure (4 and 5 kg) in our study.

2.1. Description of Data

For our study we converted each pain episode into a sequence of AUs. Each sequence is formed from the FACS codings by stringing together all AUs in order of their onset time. If two or more AUs have the same onset time—that is, appear in the same frame of the video recording—we used compounds of AUs. Within AU compounds the individual AU numbers are ordered naturally and separated by a hyphen. Note that a compound

[2]Please note that this set up was approved by an ethics committee.

au7
au6-7
au7 au4 au7 au9
au12 au19-25-26 au19-25-26 au28 au17
au6-12 au7

(a) Complete Alphabet Σ

au7
au6-7
au7 au4 au7 au9
auI auI-I-I auI-I-I auI auI
auI-6 au7

(b) Reduced Alphabet Σ_I

Figure 2: Examples of sequences. Please note that the examples over the reduced alphabet result from summarizing irrelevant AUs in the examples over the full alphabet.

Table 1: Lengths and number of distinct AUs and AU compounds separated by subject groups. Shown are the number of subjects (#sub); the number of pain episodes, or sequences, (N); the sequence length; the total number of distinct AUs and AU compounds within all sequence (#AUs); and the number of distinct AUs and AU compounds per sequence. For the sequence length the average including standard deviation ($\bar{l}$), the minimum ($\min_l$), and the maximum ($\max_l$) are given. For the number of distinct AUs and AU compounds per sequence the average including standard deviation ($\bar{t}$) and the maximum ($\max_t$) are given.

(a) Sequence counts and lengths for both alphabets

Group	#sub	N	$\bar{l}$	$\min_l$	$\max_l$
Overall	86	347	4.03 ± 3.16	1	17
Demented	31	142	4.70 ± 3.34	1	16
Healthy	55	205	3.57 ± 2.95	1	17

(b) AU count for the complete set of AUs Σ

Group	#AUs	$\bar{t}$	$\max_t$
Overall	76	3.54 ± 2.53	13
Demented	58	4.19 ± 2.66	13
Healthy	53	3.09 ± 2.34	13

(c) AU count for the reduced set of AUs Σ_I

Group	#AUs	$\bar{t}$	$\max_t$
Overall	32	2.69 ± 1.56	9
Demented	28	3.13 ± 1.68	9
Healthy	25	2.39 ± 1.40	7

AU, such as *au6-7*, denotes a single event while singular AUs succeeding each other, such as *au6 au7*, denote sequential events. Altogether, there are 76 different AUs and AU compounds, which constitute the alphabet of the to be induced grammar. However, 29 of these appear only once within the whole set of sequences. The complete list of AUs and AU compounds, sorted by frequency of occurrence, is given in Figure 1.

A sequence is a series of 1 to 17 AUs and AU compounds separated by white-space. Some example sequences can be seen in Figure 2(a). The average sequence length is 4.03 over all pain episodes (see Table 1(a)). Sequences of demented patients have on average 1.14 more AUs or AU compounds than sequences of healthy persons (unpaired t-test, $t(278.2) = 3.3$, $p = .001$). Table 1(b) gives basic statistics for the number of distinct AUs and AU compounds within sequences.

2.2. Pain-relevant Action Units

In accordance with literature, Kunz et al. [14] identified AUs occurring in more than 5% of the pain episodes (see Table 2). This increased incidence is generally used as

Table 2: AUs with an occurrence ratio of more than 5% in denominated pain episodes in healthy controls and demented patients (abbreviated from Table 1 in [14]). Relevant AUs in italics (see text).

AU	Description
au1	inner brow raiser
au2	outer brow raiser
au4	*brow lowerer*
au6	*cheek raiser*
au7	*lid tightener*
au9	*nose wrinkler*
au10	*upper lip raiser*
au17	chin raiser
au25	lips part
au26	jaw drop
au27	mouth stretch
au45	eye blink

Σ_I = {auI, au7, au4, au6, au9, auI-I, au10, au6-7, au4-7, au6-7-9, au6-9, auI-7, auI-I-I, auI-6, auI-4, au4-9, au6-7-10, au7-10, auI-9, au4-6-7, auI-6-7, au4-7-9, au4-6-7-9, au7-9, auI-10, au4-6-7-10, au6-10, auI-I-7, au4-7-10, auI-I-6-9, auI-4-7, auI-6-7-9}

Figure 3: List of AUs and AU compounds after alphabet reduction sorted by decreasing frequency of occurrence.

criterion to identify AUs which are indicative for pain [27]. Contrasting facial expressions of pain with other facial expressions, Kunz et al. [14] identified 5 AUs which are not only typical, but also relevant for pain: 4 (*brow lowerer*), 6 (*cheek raiser*), 7 (*eye lid tightener*), 9 (*nose wrinkler*), and 10 (*upper lip raiser*).

Following these findings, we reduced the number of distinct AUs. The five pain-relevant AUs are retained, all other AUs are summarized in the functional "wild card" action unit *I*. This aggregation of different tokens into an abstract representant is similar to the part-of-speech classes used in natural language processing [22]. Since there might still be information in the co-occurrence of action units, we kept the compounds—replacing the numbers of irrelevant AUs with *I*. Furthermore, since compounds denote the simultaneous occurrence of AUs, the position of the *I* within a compound must not matter. Therefore, *I*'s in compounds are moved to the front. So, the compound *au4-7-9* is kept unchanged, while the compound *au7-24* is replaced by *auI-7*. Some example sequences can be seen in Figure 2(b). This "wild card" replacement reduced the alphabet to 32 AUs and AU compounds (Figure 3). After the reduction, just 5 AU compounds appear only once within the whole set of sequences.
The reduction of the alphabet leads to a changed distribution of action units and compounds within the surveyed groups. The minimal, maximal, and average number of distinct AUs and AU compounds per sequence can be seen in Table 1(c). The sequence lengths do not change.

3. Grammar Induction

In formal language theory a grammar describes the production of words by rules. A word is a concatenation of symbols. All symbols allowed to appear in words are predefined in an alphabet and called *terminals*. Furthermore, a second set of symbols, named *non-terminals*, is given which are used as a kind of variables in production rules

which describe how words over the alphabet can be generated. Let Σ denote the finite set of terminals, also called the alphabet of the grammar, N the finite set of non-terminals, and $V = \Sigma \cup N$. Then a grammar G is formally a quadruple (N, Σ, R, S) where R is the finite set of production rules and $S \in N$ is a distinguished non-terminal called *start symbol*. A grammar is called context-free grammar (CFG) if all production rules are of the form $N \to V^*$ where * is the Kleene star operator. In the following we will focus on CFGs.

A grammar G induces the ternary relation $\overset{r}{\Longrightarrow}_G$ between strings in V^*. G derives the string y from the string x using r if x and y are identical up to a single non-terminal n in x which has been replaced according to the production rule r:

$$\begin{aligned} x \overset{r}{\Longrightarrow}_G y \quad \text{iff} \quad & G = (N, \Sigma, R, S) \wedge r \in R \wedge \\ & \exists a, b, \beta \in V^*, n \in N : \big(\quad (x = a.n.b) \wedge \\ & \qquad\qquad\qquad\quad (y = a.\beta.b) \wedge \\ & \qquad\qquad\qquad\quad (r = (n \to \beta)) \big) \end{aligned}$$

where "." denotes string concatenation.

In the ternary relation the rule r used to derive y from x is given explicitly. If only the derives-relation but not the used rule is of interest, this information can be captured in the binary relation $\Rightarrow_G$ between strings in V^*. G derives the string y from the string x if there exists a production rule in R such that $x \overset{r}{\Longrightarrow}_G y$. Let the binary relation $\Rightarrow_G^*$ be the reflexive transitive closure of $\Rightarrow_G$. Then a concatenation of terminals w is called *word* of the grammar if and only if $S \Rightarrow_G^* w$. The set of all words derivable from the start symbol is called the language of G, $\mathbf{L}(G) = \{w \in \Sigma^* | S \Rightarrow_G^* w\}$.

Typically, grammar formalisms are used to characterize classes of languages and to derive generators and acceptors for such languages. In the context of machine learning, the inverse process is of interest: the induction of a grammar from sample words. Approaches to grammar inference are present since the early research on formal grammars [5, 12, 31].

3.1. Alignment-Based Learning

Alignment-Based Learning[3] is an unsupervised grammar inference algorithm proposed by van Zaanen [33, 34, 35] which learns context-free grammars from sample words. The algorithm is based on the idea that symbol groups within a word have inherent semantic meaning like morphemes in natural languages. These symbol groups are called constituents. The goal of the algorithm is to find the best constituents for a given set of words. This is achieved in two phases: the alignment phase and the selection learning phase.

In the *alignment phase* the sample words are compared pairwise. Identical symbols are aligned between the words. The *differing symbols* appear within the same context and, thus, it is possible that both symbol groups form a morpheme each and interchanging these is grammatical. Consider the following two English sentences:

John likes green apples.
John likes red apples.

[3] An implementation of ABL is available under `http://ilk.uvt.nl/menno/research/software/abl`.

In our notion both sentences are words build from uninflective symbols, like *apples*. We see that both words are identical besides the symbols *green* and *red*, respectively. Thus, these symbols form a possible constituent and the words are annotated accordingly:

John likes $(\text{green})_1$ apples.
John likes $(\text{red})_1$ apples.

This process is repeated iteratively with the original words and the newly annotated words. This allows for nestings of constituents. If, for example, the second word is compared with the new word "John likes baseball" the following constituents are formed:

John likes $((\text{green})_1 \text{ apples})_2$.
John likes $((\text{red})_1 \text{ apples})_2$.
John likes $(\text{baseball})_2$.

In general, the alignments might not be unique, that is, there might be overlapping candidates for constituents. Thus, in the *selection learning phase* these possible candidates for constituents are reduced using frequency information. The possibility of each possible constituent is calculated and for each word the constituent combination with the highest probability is kept.

Finally, the constituents are transformed into a grammar: each constituent is marked with a previously unused non-terminal and for each possible replacement in the constituent a production rule is created. So, for the first constituent in the example above the non-terminal n_1 would be created along with the production rules $n_1 \to$ green, and $n_1 \to$ red. The second constituent would form the non-terminal n_2 with the production rules $n_2 \to n_1$ apples, and $n_2 \to$ baseball.

3.2. Heuristic Rule Extraction

Alignment-Based Learning might result in grammars with a high number of rules, many of which might cover (sub-) words which only occur once in a sample of words. In the domain of sequences of facial expressions, it might be the case that the occurrences of some sub-words of AUs are highly person specific. To obtain structural patterns which characterize *typical* expression sequences, it is desirable to remove such rules from the grammar.

Let us define a stochastic context-free grammar (SCFG) as quintuple $G = (N, \Sigma, R, S, P)$ where N, Σ, R, and S have the same meaning as for CFGs. P is a total function from R to $[0, 1]$ such that

$$\forall X \in N : \left[\sum_{\beta \in V^*} P(X \to \beta) \right] = 1.$$

As first step we modify ABL's transformation of constituents to production rules to yield a SCFG: For each constituent the frequency of its different replacement possibilities β_i is counted. When the previously unused non-terminal n is introduced along with its production rules, the probability of each production rule $n \to \beta_i$ is the relative frequency of the replacement β_i within the constituent:

$$P(n \to \beta_i) = \frac{\mathrm{h}_n(\beta_i)}{\sum_j \mathrm{h}_n(\beta_j)} ,$$

Input: a SCFG $G = (N, \Sigma, S, R, P)$; a threshold t
Output: a CFG with reduced rule set

initialize queue Q;
add triple $(S, \emptyset, 1)$ to Q;
$R' \leftarrow \emptyset$

while *Q is not empty* **do**
  pop (s, r, p) from Q;
  if $p > t$ **then**
    if $s \in \Sigma^*$ **then**
      $R' \leftarrow R' \cup r$;
    else
      foreach $r' \in R$ *such that* $s \overset{r'}{\Longrightarrow} s'$ **do**
        push $(s', r \cup \{r'\}), p \times \mathrm{P}(r'))$;
      end
    end
  end
end
return (N, Σ, S, R');

Algorithm 1: Heuristic rule extraction algorithm

where $\mathrm{h}_n(\beta_j)$ denotes the count of the replacement possibility β_j within the constituent transformed into n.

Continuing the above example, the rules $n_1 \rightarrow$ green, and $n_1 \rightarrow$ red both have a probability of .5 since within the first constituent the replacements *green* and *red* each occur once. On the other hand, $P(n_2 \rightarrow n_1$ apples$) = 2/3$ and $P(n_2 \rightarrow$ baseball$) = 1/3$, since the replacement n_1 *apples* occurs twice while *baseball* occurs only once.

Any grammar constructed by this ABL modification potentially contains production rules which have a probability of 1. Such trivial rules are eliminated as further post-processing step: while R contains some rule $r_t = n_t \rightarrow \beta_t$ such that $P(r_t) = 1$ and $n_t \neq S$, remove n_t from N, remove r_t from R, and replace each remaining rule $n \rightarrow \beta$ in R by $n \rightarrow \beta'$, where $\beta \overset{r_t}{\Longrightarrow}_G \beta'$.

Let the probability $P_G(\beta, (r_1, \ldots, r_m))$ to derive some $\beta \in V^*$ from the S, the start symbol of G, by a sequence of rules be defined by the following recursive definition:

$$\mathrm{P}_G\left(\beta, ()\right) = \begin{cases} 1 & \text{if } \beta = S \\ 0 & \text{otherwise} \end{cases}$$

$$\mathrm{P}_G\left(\beta_m, (r_1, \ldots, r_m)\right) = \begin{cases} \mathrm{P}_G\left(\beta_{m-1}, (r_1, \ldots, r_{m-1})\right) P(r_m) & \text{if } \beta_{m-1} \overset{r_m}{\Longrightarrow}_G \beta_m \\ 0 & \text{otherwise} \end{cases}$$

Heuristic rule extraction reduces the number of rules in a grammar while preserving the most frequent words of the language. If a word can be derived from the start symbol with a sequence of rules $(r_1, \ldots, r_m)$ with a probability higher than a given threshold t, the production rules used in the derivation are memorized. The memorized rules, the terminals, the non-terminals, and the start symbol form the reduced CFG. The complete

algorithm can be found in Algorithm 1.

3.3. Evaluation Methodology

While there is an established approach for evaluating the quality of learned hypotheses in standard machine learning, this is not true for unsupervised empirical grammar inference [36]. One method proposed by van Zaanen and Geertzen [36] is to measure the ability of the learned grammar to classify sequences based on language membership: Some words from the target language are used to induce a grammar G. Then this grammar is tested on words from the same language and words not in the language. All words from the same language should be derivable by G. At the same time, no word which is not in the language should be derivable by G. The crucial issue of this method is choosing appropriate sample words which are not in the language.

In the context of supervised learning, the performance of a learning algorithm can be estimated by k-fold cross-validation [23]: Given a set of training data D which is pre-classified in positive and negative examples, this set is divided into k distinct subsets $(D_i)_{1\leq i\leq k}$, also called folds. For each subset D_j a classifier is trained from the other subsets $\bigcup_{1\leq i\leq k, i\neq j} D_i$. The performance of this classifier is evaluated on the subset D_j. The overall performance is the average of the performance measures over the subsets.

We propose to apply cross-validation to evaluate the quality of the learned pain grammars. Words from the pain episodes are positive examples D^+, words from a randomly generated different language form the negative examples D^-. Both sets are divided in k distinct subsets, $\left(D_i^+\right)_{1\leq i\leq k}$ and $\left(D_i^-\right)_{1\leq i\leq k}$. For each run j we induce a grammar G_j from $\bigcup_{1\leq i\leq k, i\neq j} D_i^+$. Then we calculate recall, precision, and accuracy on the positive and negative examples of the subset $D_j^+ \cup D_j^-$. *True positives* are words from D_j^+ which can be derived by G_j. *False negatives* are words from D_j^+ which cannot be derived by G_j. *True negatives* are words from D_j^- which cannot be derived by G_j. *False positives* are words from D_j^- which can be derived by G_j. Then recall, precision, and accuracy for a fold j can be calculated the usual way:

$$\begin{aligned}
\text{precision}_j &= \frac{\text{true positives}}{\text{true positives} + \text{false positives}} = \frac{\left|\left\{w \in D_j^+ \mid S \Rightarrow_G^* w\right\}\right|}{\left|\left\{w \in \left(D_j^+ \cup D_j^-\right) \mid S \Rightarrow_G^* w\right\}\right|} \\
\text{recall}_j &= \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} = \frac{\left|\left\{w \in D_j^+ \mid S \Rightarrow_G^* w\right\}\right|}{\left|D_j^+\right|} \\
\text{accuracy}_j &= \frac{\text{true positives} + \text{true negatives}}{\text{true positives} + \text{false positives} + \text{true negatives} + \text{false negatives}} \\
&= \frac{\left|\left\{w \in D_j^+ \mid S \Rightarrow_G^* w\right\}\right| + \left|\left\{w \in D_j^- \mid \neg\left(S \Rightarrow_G^* w\right)\right\}\right|}{\left|D_j^+ \cup D_j^-\right|}
\end{aligned}$$

The overall performance is given by the averages of precision, recall, and accuracy over all folds.

Besides evaluating the quality of grammar learning, we are interested in obtaining a grammar for further psychological research, e.g., to compare classification accuracies between human raters and automated classification systems. The learned grammar should be as effective and efficient as possible, i.e., have as few rules as possible while having high performance. Therefore, we analyze whether a learned grammar contains rules

Table 3: Description of positive examples separated for 10-fold cross-validation. The number of sequences, or words, (N) were observed on #sub different subjects. For the word length the average including standard deviation ($\bar{l}$), the minimum ($\min_l$), and the maximum ($\max_l$) are given.

Group	#sub	N	$\bar{l}$	$\min_l$	$\max_l$
Overall	86	347	4.03 ± 3.16	1	17
Fold 1	24	34	5.76 ± 3.75	1	17
Fold 2	29	35	4.00 ± 2.92	1	13
Fold 3	27	35	3.54 ± 3.00	1	16
Fold 4	26	35	4.03 ± 2.86	1	11
Fold 5	28	35	3.43 ± 2.97	1	11
Fold 6	31	34	4.09 ± 3.55	1	13
Fold 7	29	35	3.51 ± 2.96	1	13
Fold 8	30	34	3.91 ± 3.07	1	11
Fold 9	28	35	3.97 ± 3.23	1	15
Fold 10	28	35	4.11 ± 3.03	1	13

which are only necessary for a small number of words. If there are such rules it should be possible to reduce the number of rules in a grammar significantly while still keeping high performance. We use the heuristic rule extraction method proposed above to obtain such a reduced grammar. First we learn a grammar G over all words from pain episodes. Then we reduce the rule set of this grammar by heuristic rule elimination using succeedingly higher thresholds yielding G_t's. Finally we assess recall, precision, and accuracy of the reduced grammars on words from pain episodes and randomly generated words. No cross-validation is performed for this analysis.

4. Application and Results

In order to have positive examples for cross-validation we divided the empirical pain words described in Section 2 in 10 folds. The words were assigned randomly to the folds such that all folds have approximately the same number of words. However, the folds have an equal ratio of words from demented patients and healthy persons. The number of subjects which contributed to each fold, the total number of words, and basic statistics on the word lengths can be seen in Table 3.

Each fold is extended by negative examples which are randomly generated words under the following constraints: For each positive example exactly one negative example is constructed. For each fold, the distribution of word lengths and the terminal frequency within negative examples are identical to those within the positive examples. For the complete and for the reduced alphabet, the total number of distinct terminals and their distribution among the words are shown in Table 4. Since all folds contain positive examples of only one terminal and the occurrence frequency of some terminals, such as *au7* or *auI*, is high, the negative examples inevitably contain words which are also contained in the positive examples. This might lead to an underestimation of the performance. On the other hand, identical words can occur more than once in our sample. Therefore, the same word might be seen during training and test, what leads to an overestimation of the performance.

Table 4: Number of distinct AUs and AU compounds separated for 10-fold cross-validation. Shown are the total number of distinct terminals in a fold (#AUs), the average number of distinct terminals per word including standard deviation ($\bar{t}$), and the maximum number of distinct terminals per word ($\max_t$).

(a) Complete Alphabet Σ

		Positive Examples		Negative Examples	
Group	#AUs	$\bar{t}$	$\max_t$	$\bar{t}$	$\max_t$
Overall	76	3.54 ± 2.53	13	3.38 ± 2.33	13
Fold 1	36	4.94 ± 3.05	13	4.59 ± 2.75	13
Fold 2	30	3.46 ± 2.20	9	3.37 ± 2.06	9
Fold 3	31	3.23 ± 2.57	13	2.91 ± 1.90	13
Fold 4	33	3.49 ± 2.21	9	3.37 ± 2.26	9
Fold 5	31	3.00 ± 2.33	9	2.80 ± 2.07	9
Fold 6	28	3.74 ± 3.04	10	3.38 ± 2.41	10
Fold 7	31	3.17 ± 2.26	9	3.06 ± 2.29	9
Fold 8	30	3.44 ± 2.57	9	3.44 ± 2.52	9
Fold 9	34	3.49 ± 2.54	10	3.37 ± 2.43	10
Fold 10	31	3.51 ± 2.24	9	3.57 ± 2.32	9

(b) Reduced Alphabet Σ_I

		Positive Examples		Negative Examples	
Group	#AUs	$\bar{t}$	$\max_t$	$\bar{t}$	$\max_t$
Overall	32	2.69 ± 1.56	9	2.60 ± 1.53	9
Fold 1	15	3.41 ± 1.52	7	3.21 ± 1.65	7
Fold 2	13	2.77 ± 1.44	6	2.71 ± 1.36	6
Fold 3	14	2.71 ± 1.89	9	2.49 ± 1.46	9
Fold 4	16	2.66 ± 1.43	6	2.49 ± 1.36	6
Fold 5	16	2.40 ± 1.70	7	2.37 ± 1.54	7
Fold 6	13	2.76 ± 1.78	6	2.53 ± 1.58	6
Fold 7	14	2.37 ± 1.35	5	2.31 ± 1.47	5
Fold 8	14	2.56 ± 1.58	6	2.53 ± 1.71	6
Fold 9	15	2.71 ± 1.45	6	2.74 ± 1.63	6
Fold 10	16	2.60 ± 1.38	5	2.60 ± 1.48	5

A → auI .115 | au7 .081 | auI AD .043 | au4 .035 | I E .032 | F H .020 | C E .020 |
C H .020 | C CB .020 | JE H .014 | BE H .014 | C auI AD .014 | au4 E .012 |
auI-I .012 | F AD .012 | FC H .012 | au6-7 .009 | I AG .009 | FC E .009 | ...

B → auI J .392 | auI .235 | au4 .078 | au7 .039 | au6 .039 | IB HD .039 | au10 .020 |
au7 H .020 | auI-I HD .020 | AC J .020 | IB E .020 | GAB CB .020 | auI GI .020 |
IB CB .020 | IB AG .020

C → au7 .133 | au4 .117 | auI-I .117 | au6 .083 | au6-7-9 .067 | au6-7 .050 | I ED .050 |
F CD .050 | auI-6-7 .033 | au4-7 .033 | JE CD .033 | au6-7-10 .017 | ...

E → au7 .200 | auI .185 | auI AD .092 | auI B .077 | au9 .062 | au10 .046 | FH H .046 |
auI J .046 | au6 .031 | au6-7 .015 | au6-7-9 .015 | auI-10 .015 | au4 .015 | ...

Figure 4: Excerpt of the grammar induced for the complete set of words using the reduced alphabet. For each non-terminal multiple replacement alternatives with their probabilities are shown (separated by "|"). The replacement alternatives are ordered by decreasing probability. The non-terminal A is the start symbol.

Table 5: Precision, recall, and accuracy achieved in the 10-fold cross-validation. Shown are average and standard deviation.

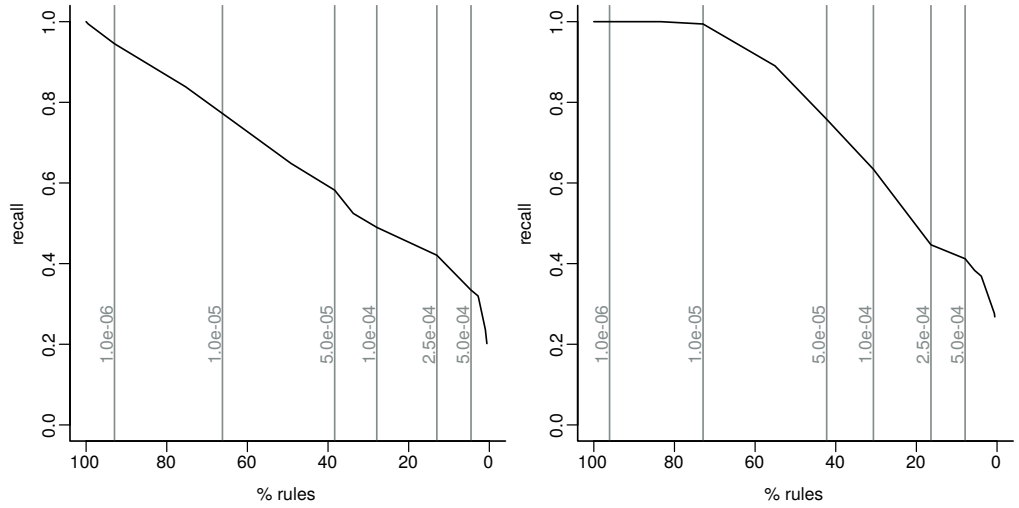
Alphabet	precision	recall	accuracy
Σ	.530±.022	.645±.071	.537±.028
Σ_I	.504±.016	.914±.038	.506±.029

We used 10-fold cross-validation as described in Section 3.3 to evaluate the performance. The grammars were induced separately for each of the 10 runs using ABL. To identify which words can be derived by the induced grammar we used DParser[4] which constructs an acceptor for a given grammar.

Performance values for both alphabets are displayed in Table 5. We conducted paired t-tests to compare the performance in complete alphabet and reduced alphabet conditions. Reducing the alphabet significantly increases the recall by over 25 percent points ($t(13.7) = 10.5$, $p < .001$). The effects on precision and accuracy are reversed, but also smaller. Reducing the alphabet decreases the precision by just over 2.5 percent points ($t(16.8) = 3.0$, $p = .007$). Accuracy is decreased by just over 3 percent points ($t(18.0) = 2.5$, $p = .02$).

For both alphabets the grammars induced over the complete samples are rather complex with 1322 production rules for Σ and 897 production rules for Σ_I. An extract of the induced grammar for Σ_I is given in Figure 4. Heuristic rule extraction leads to the expected reduction of rules which is summarized in Figure 5 and given in detail for some thresholds in Table 6. Overall, recall and accuracy drop with higher thresholds. Initially, recall is dropping slower than the number of rules for the reduced alphabet. Afterward, recall and number of rules are linearly dependent by trend which implies that the grammar for Σ_I contains some irrelevant rules. The grammar for the complete alphabet contains only a few irrelevant production rules since recall and number of rules drop simultaneously.

[4] `http://dparser.sourceforge.net/`



(a) Complete Alphabet Σ (b) Reduced Alphabet Σ_I

Figure 5: Recall drop induced by reduction of rules. Shown is the recall of reduced grammars constructed via heuristic rule extraction for differing thresholds by the percentage of retained production rules. Vertical lines depict selected threshold values.

Nevertheless, the results show that the aggregation of irrelevant action units, i.e., using the reduced alphabet Σ_I, improves the recall—both in the cross-validation and in the rule reduction experiment.

5. Conclusion

We presented a first exploration whether grammar inference is apt to discover underlying sequential regularities in facial pain expressions. Using ABL as an out of the box grammar inference system, we could show that grammars which describe the general structure of pain sequences do generalize reasonably well to unseen pain sequences.

The ABL algorithm is one of many grammar induction algorithms. In further studies we will compare different approaches to learn generalized sequences. We already investigated genetic algorithms in a preliminary study [32]. In addition, another algorithm for learning the SCFG for heuristic rule extraction could be employed [4].

In this study we decided to represent pain sequences by the sequence of the *onset times* of *all* AUs. By combining different AUs to a single symbol we changed the second aspect of this representation. Further studies varying the representation should be conducted. First of all, a systematic evaluation of the retained AUs can be done. Secondly, some AUs might be ignored completely, instead of being combined. Additionally, the onset and offset times might be used to represent the sequences, though the diagnostically relevant information is mostly in the onset of an AU—which is rather fast and then the intensity is slowly fading.

Another aspect of the representation is the combination of simultaneously occurring AUs to AU compounds. This leads to terminals with low occurrence frequency. These

Table 6: Results of grammar reduction with heuristic rule extraction. For select thresholds the number of retained rules after reduction (#rules) and the percentage of retained rules within the complete rule set (rules%) is shown. Furthermore, recall, precision, and accuracy calculated with the retained rules are given.

(a) Complete Alphabet Σ

threshold	#rules	rules%	recall	precision	accuracy
1e-07	1315	99.5	.994	.610	.679
1e-06	1229	93.0	.945	.599	.656
1e-05	875	66.2	.772	.600	.628
1e-04	369	27.9	.490	.574	.563
1e-03	35	2.6	.314	.562	.535
1e-02	8	.6	.202	.603	.535

(b) Reduced Alphabet Σ_I

threshold	#rules	rules%	recall	precision	accuracy
1e-07	895	99.8	1.000	.521	.540
1e-06	862	96.1	1.000	.521	.540
1e-05	654	72.9	.994	.527	.550
1e-04	275	30.7	.634	.525	.530
1e-03	35	3.9	.369	.516	.512
1e-02	5	.6	.268	.528	.514

rare terminals are a challenge for grammar inference algorithms. This difficulty can be evaded by representing sequences as sets of logic formulas. Then, the occurrences of action units are events, which can happen after another or sequentially. Also, onset and offset times can be explicitly represented as time intervals. From Allen's interval calculus [1] we know that pairs of intervals can stand in one of seven relations, such as *before*, *includes*, or *overlaps*. Given this logical representation, generalizations over pain episodes can be calculated using inductive logic programming [24].

Regarding the importance of sequence information in facial expressions, we ran a first psychological experiment [30]: We created short video sequences of an avatar showing facial expressions of pain and disgust. In the base condition, the avatar shows all modeled AUs simultaneously, that is, all AUs start at the same time and finish at the same time. In the sequential condition, the AUs start in some sequence and finish in the same sequence. Results show that subjects give more correct classifications for sequential videos in contrast to the simultaneous videos. Furthermore, the sequential presentation of AUs was rated as more natural than the simultaneous presentation.

Acknowledgments

This study was supported by the Dr. Werner Jackstädt-Stiftung. Thanks to Jirka Lewandowski who conducted some initial analyzes. Special thanks to Menno van Zaanen who willingly answered our questions about ABL. Finally, we want to thank two anonymous reviewers whose comments and suggestions helped to significantly improve this paper.

References

[1] Allen, J. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.

[2] Ashraf, A. B., Lucey, S., Cohn, J. F., Chen, T., Ambadar, Z., Prkachin, K. M., and Solomon, P. E. (2009). The painful face – pain expression recognition using active appearance models. *Image Vision Comput.*, 27(12):1788–1796.

[3] Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer, New York.

[4] Charniak, E. (1993). *Statistical language learning*. MIT Press, Cambridge, MA.

[5] Chomsky, N. (1956). Three models for the description of language. In *IRE Transactions on Information Theory, Vol. IT-2, Proceedings of the Symposium on Information Theory*, pages 113–124.

[6] Cohn, J. F. (2010). Advances in behavioral science using automated facial image analysis and synthesis. *IEEE Signal Processing Magazine*, 128(6):128–133.

[7] de la Higuera, C. (2005). A bibliographical study of grammatical inference. *Pattern Recognition*, 38(9):1332–1348.

[8] de la Higuera, C., Oates, T., and van Zaanen, M. (2008). Introduction: Special issue on applications of grammatical inference. *Applied Artificial Intelligence*, 22(1-2).

[9] Ekman, P. and Friesen, W. V. (1978). *The Facial Action Encoding System: A Technique for the Measurement of Facial Movement*. Consulting Psychologists Press, Palo Alto, CA.

[10] Fasel, B. and Luettin, J. (2003). Automatic facial expression analysis: a survey. *Pattern Recognition*, 36(1):259–275.

[11] Glowacka, D., Shawe-Taylor, J., Clark, A., de la Higuera, C., and Johnson, M. (2011). Introduction to the special topic on grammar induction, representation of language and language learning. *Journal of Machine Learning Research*, 12:1425–1428.

[12] Gold, E. (1967). Language identification in the limit. *Information and Control*, 10:447–474.

[13] Han, J., Kamber, M., and Paei, J. (2006). *Data mining: concepts and techniques*. Morgan Kaufman.

[14] Kunz, M., Scharmann, S., Hemmeter, U., Schepelmann, K., and Lautenbacher, S. (2007). The facial expression of pain in patients with dementia. *Pain*, 133:221–228.

[15] Lajevardi, S. M. and Hussain, Z. M. (2012). Automatic facial expression recognition: feature extraction and selection. *Signal, Image and Video Processing*, 6(1):159–169.

[16] Lenat, D. B. (1983). The role of heuristics in learning by discovery: Three case studies. In *Machine learning*, pages 243–306. Springer.

[17] Lien, J.-J. J., Kanade, T., Cohn, J., and Li, C.-C. (1998). Automated facial expression recognition based on facs action units. In *Third IEEE International Conference on Automatic Face and Gesture Recognition*, pages 390–395.

[18] Lints-Martindale, A. C., Hadjistavropoulos, T., Barber, B., and Gibson, S. J. (2007). A psychophysical investigation of the facial action coding system as an index of pain variability among older adults with and without Alzheimer's disease. *Pain Medicine*, 8(8):678–689.

[19] Lucey, P., Cohn, J. F., Matthews, I., Lucey, S., Sridharan, S., Howlett, J., and Prkachin, K. M. (2011). Automatically detecting pain in video through facial action units. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 41(3):664–674.

[20] Maimon, O. Z. and Rokach, L. (2005). *Data mining and knowledge discovery handbook*, volume 1. Springer.

[21] Malatesta, L., Karpouzis, K., and Raouzaiou, A. (2009). Affective intelligence: The human face of ai. In *Artificial Intelligence: An International Perspective*, volume 5640 of *LNCS*, pages 53–70. Springer.

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

[23] Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, New York.

[24] Muggleton, S. (1996). Learning from positive data. In Muggleton, S., editor, *Proceedings of the 6th International Workshop on Inductive Logic Programming*, pages 225–244. Stockholm University, Royal Institute of Technology.

[25] Paliouras, G. and Sakakibara, Y. (2007). Guest editorial to the special issue on grammatical inference. *Machine Learning*, 66(1):3–5.

[26] Parekh, R. and Honavar, V. (2000). Automata induction, grammar inference, and language acquisition. In Dale, R., Somers, H. L., and Moisl, H., editors, *Handbook of Natural Language Processing*, pages 746–785. Marcel Dekker, New York.

[27] Prkachin, K. M. (1992). The consistency of facial expressions of pain: a comparison across modalities. *Pain*, 51:297–306.

[28] Schmid, U. and Kitzelmann, E. (2011). Inductive rule learning on the knowledge level. *Cognitive Systems Research*, 12(3):237–248.

[29] Schmid, U., Siebers, M., Seuß, D., Kunz, M., and Lautenbacher, S. (2012). Applying grammar inference to identify generalized patterns of facial expressions of pain. In Heinz, J., de la Higuera, C., and Oates, T., editors, *Proceedings of the Eleventh International Conference on Grammatical Inference, ICGI 2012, University of Maryland, College Park, USA, September 5-8, 2012*, pages 183–188.

[30] Siebers, M., Engelbrecht, T., and Schmid, U. (2013). On the relevance of sequence information for decoding facial expressions of pain and disgust – An avatar study. In Reichardt, D., editor, *Proceedings of the 7th Workshop on Emotion and Computing - Current Research and Future Impact (KI 2013)*, pages 3–9.

[31] Solomonoff, R. J. (1959). A progress report on machines to learn to translate languages and retrieve information. In *Advances in Documentation and Library Science*, volume III Part 2, pages 941–953, New York, London. Interscience Publishers.

[32] Stocker, C., Siebers, M., and Schmid, U. (2013). Erkennung von sequenzen mimischer schmerzausdrücke – ein genetischer algorithmus. In Henrich, A. and Sperker, H.-C., editors, *LWA 2013. Lernen, Wissen & Adaptivität. Workshop Proceedings*, pages 117–120.

[33] van Zaanen, M. (2002a). *Bootstrapping Structure into Language: Alignment-Based Learning*. PhD thesis, University of Leeds, UK.

[34] van Zaanen, M. (2002b). Implementing alignment-based learning. In Adriaans, P. W., Fernau, H., and van Zaanen, M., editors, *Grammatical Inference: Algorithms and Applications, 6th International Colloquium (ICGI 2002)*, volume 2484 of *LNCS*, pages 312–314. Springer.

[35] van Zaanen, M. (2003). Theoretical and practical experiences with alignment-based learning. In *Proceedings of Australasian Language Technology Workshop (ALTW2003)*, pages 25–32.

[36] van Zaanen, M. and Geertzen, J. (2008). Problems with evaluation of unsupervised empirical grammatical inference systems. In Clark, A., Coste, F., and Miclet, L., editors, *Grammatical Inference: Algorithms and Applications, 9th International Colloquium (ICGI 2008)*, volume 5278 of *LNCS*, pages 301–303. Springer.