BioMedInformatics

MDPI

Article

Ensemble of HMMs for Sequence Prediction on Multivariate Biomedical Data

Richard Fechner [1,2], Jens Dörpinghaus [2,3,4,*], Robert Rockenfeller [5,6,7,*] and Jennifer Faber [4,8,9]

1 Eberhard Karls University Tübingen, 72074 Tübingen, Germany
2 Federal Institute for Vocational Education and Training (BIBB), 53113 Bonn, Germany
3 Department of Computer Science, University of Koblenz, 56070 Koblenz, Germany
4 German Center for Neurodegenerative Diseases (DZNE), 53127 Bonn, Germany
5 Mathematical Institute, University of Koblenz, 56070 Koblenz, Germany
6 School of Science, Technology and Engineering, University of the Sunshine Coast, Maroochydore, QLD 4558, Australia
7 School of Biomedical Sciences, University of Queensland, Brisbane, QLD 4072, Australia
8 Center for Neurology, Department of Parkinson, Sleep and Movement Disorders, University Hospital Bonn, 53127 Bonn, Germany
9 Department of Neuroradiology, University Hospital Bonn, 53127 Bonn, Germany
* Correspondence: doerpinghaus@uni-koblenz.de (J.D.); rrockenfeller@uni-koblenz.de (R.R.)

Abstract: Background: Biomedical data are usually collections of longitudinal data assessed at certain points in time. Clinical observations assess the presences and severity of symptoms, which are the basis for the description and modeling of disease progression. Deciphering potential underlying unknowns from the distinct observation would substantially improve the understanding of pathological cascades. Hidden Markov Models (HMMs) have been successfully applied to the processing of possibly noisy continuous signals. We apply ensembles of HMMs to categorically distributed multivariate time series data, leaving space for expert domain knowledge in the prediction process. **Methods:** We use an ensemble of HMMs to predict the loss of free walking ability as one major clinical deterioration in the most common autosomal dominantly inherited ataxia disorder worldwide. **Results:** We present a prediction pipeline that processes data paired with a configuration file, enabling us to train, validate and query an ensemble of HMMs. In particular, we provide a theoretical and practical framework for multivariate time-series inference based on HMMs that includes constructing multiple HMMs, each to predict a particular observable variable. Our analysis is conducted on pseudo-data, but also on biomedical data based on Spinocerebellar ataxia type 3 disease. **Conclusions:** We find that the model shows promising results for the data we tested. The strength of this approach is that HMMs are well understood, probabilistic and interpretable models, setting it apart from most Deep Learning approaches. We publish all code and evaluation pseudo-data in an open-source repository.

Keywords: longitudinal modeling; HMM; prediction pipeline; biomedical data prediction

Citation: Fechner, R.; Dörpinghaus, J.; Rockenfeller, R.; Faber, J. Ensemble of HMMs for Sequence Prediction on Multivariate Biomedical Data. *BioMedInformatics* **2024**, *4*, 1672–1691. https://doi.org/10.3390/biomedinformatics4030090

Academic Editors: Jörn Lötsch and Alexandre G. De Brevern

Received: 27 March 2024
Revised: 15 May 2024
Accepted: 3 June 2024
Published: 3 July 2024

1. Introduction

Probabilistic Models are used to infer latent states from signals. In the context of health and disease, observational signs and symptoms represent such measurable signals, whilst the underlying state of the disease may be modeled as latent. Here, a central question is the identification of temporal order and predictive features for deterioration. Several studies have investigated particular diseases like Parkinson's disease or ADHD; see for example [1–3].

However, there are specific challenges in inferring unknown quantities from observation, for example due to missing criteria or guidelines [2]. In addition, data and time-series often are incomplete, biased or may contain errors [4,5]. Generally, solutions are developed for one particular research scenario, but in healthcare scenarios it is not clear how quantities influence the progress of a disease [6–8].

The core objectives of this paper concern the last question: The methodological novelty presented in this article is the ability to include domain knowledge in the prediction by a re-weighting of the individual constituents of the ensemble. The proposed framework also allows for automatic hyper-parameter tuning in case domain knowledge is sparse. The flexibility of the model enables the quick prototyping of a model without re-training (see Figure 1). Furthermore, our work extends this conceptual principle to a more general case, in which the model allows for maximum flexibility and generality by being able to choose any constellation of different observable variables to infer knowledge about any other observable variable. This extension is realized within the widely used probabilistic framework of Hidden Markov Models (HMMs). Our work presents two test environments, including the application of the model to real-world medical data based on SCA3.

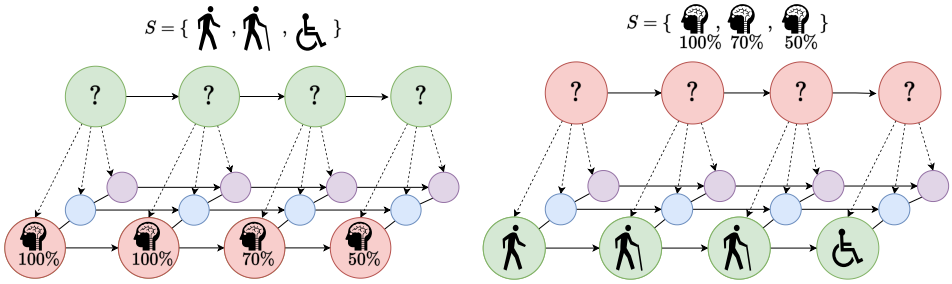

Figure 1. Conceptual depiction of the flexibility enabled by the proposed model. The predicted variable as well as the means by which to make a prediction are chosen freely by the user.

Remark on the choice of model: Studies run over a certain time horizon, continuously producing data—sometimes in irregular time intervals. A model type is needed that easily incorporates past knowledge whilst being able to handle missing data and retaining interpretability at every time step. The proposed model falls within this probabilistic model class. We propose an ensemble of Hidden Markov Models (HMMs) that are able to update their priors with every new data-point received. We think it is critical to understand our motivation behind the choice of Hidden Markov Models for this specific application on biomedical multivariate time-series data, even more so, as Deep Learning approaches gain a significant amount of attention. We would like to note that unlike Deep Learning models, HMMs are *explanatory* models, making them a solid choice for applications in medicine where one would like to understand the workings of the ML algorithm. Furthermore, HMMs are probabilistic generative models, increasing the range of downstream use-cases (e.g., data augmentation). Another constraint in the choice of the model is the limitations imposed by the medical studies themselves. If not enough data are provided to train a Neural Network, we are faced with class imbalances and missing values. The generic and flexible framework also allows us to not only consider a hidden state, as is usually done with HMMs, **but any observable state** as the predicted or output variable of our inference. We generate a strong prior by training the HMMs in a supervised fashion, maintaining the ability to train the model unsupervised down the line.

We present a generic prediction pipeline that processes data paired with a configuration file, enabling to construct, validate and query a fully parameterized HMM-based model. Specifically, we will propose a rather new technique that includes constructing multiple HMMs—one for each variable sequence of the multivariate sequence—to predict another observable variable. We will demonstrate in a use case that the newly proposed technique can perform well when tested on real-world application data. Finally, we will discuss the strengths and weaknesses of the model and how the approach could be augmented to improve results

The aim of this article is thus to (a) provide a theoretical and practical framework for multivariate time-series inference based on HMMs and (b) apply and validate the proposed prediction pipeline (see Figure 2) to real-world data. This paper is divided into six sections. The first section provides an introduction, a brief overview on the real-word use case

and related work. A second section provides the methodological background. The third section describes the novel methodological approach, the data and pipeline. The fourth section is dedicated to experimental results and the evaluation of this novel approach. A detailed discussion is presented in section five. Our conclusions and outlooks are drawn in the final section. We will now briefly discuss the background of our real-world scenario and usecase.

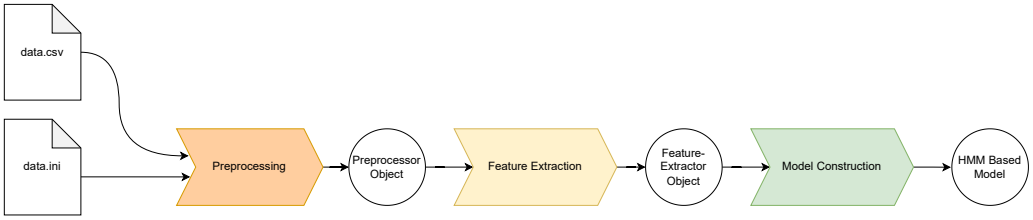

Figure 2. Concept of the presented prediction pipeline, allowing for HMM-based model construction and model query.

1.1. Use Case

The real-world data we present for the use case here are observational data from large natural history studies in the most common autosomal dominantly inherited ataxia disorder worldwide, spinocerebellar ataxia type 3, see [9–12]. Spinocerbellar ataxia 3 is a neurodegenerative disease with onset of symptoms in adult life, around the fourth decade, see [13,14]. Clinical hallmarks are a progressive loss of balance, coordination deficits and slurred speech. SCA3 patients experience significant restrictions of mobility and communicative skills. Notably, SCA3 is a so-called rare disease with a prevalence of <3 per 100,000. Ataxia, as the most prominent symptom, is measured with a scale that assesses eight different items, like gait, stance or speech. In the observational studies included here, participants were assessed on an annual basis and the resulting scores (itemwise as well as the overall ataxia severity sum score) are considered as signals. Measuring signals for consecutive discrete time steps present so-called multivariate time-series or—to put it in other words—a series of multiple observable variables over a period of time. Our aim is to infer knowledge about a state-sequence of an unknown variable from these sequences of observations. Since HMMs have been widely used within biomedical research, we will now briefly discuss related work to show the novelty of our approach.

1.2. Related Work

HMMs have been successfully applied to the processing of possibly noisy continuous signals in the case of speech or gesture recognition [15–17], as well as to the processing of sequences of discrete signals like text categorization [18]. In the context of biomedical data, they have been incorporated into forecast models, improving estimates about patient mortality [19]. Here, we will focus on multivariate time-series of categorically distributed data.

These approaches have been mainly used for classification [20,21], clustering [22,23] and anomaly detection [24]. Other approaches combined HMMs with fuzzy models [25] or latent variables [26]. However, our novel approach presents a model able to choose any constellation of different observable variables to infer knowledge about any other observable variable. This is related to knowledge graph approaches [27–29], in particular in the biomedical field [30,31]. This—together with the proposed implementation of a generic workflow—clearly shows the novelty of our contribution. We will now continue with the methodological background of our work.

2. Background

2.1. Markov Chains

A first-order **Markov Chain** is a stochastic process describing the transition between a system's states.

Definition 1. *A Markov Chain is defined by the set of N states $S = \{S_1, ..., S_N\}$ and a $N \times N$ row-stochastic transition matrix $\mathcal{A}$, where a_{ij} defines the probability of transitioning from state S_i to state S_j in a single time step.*

We will be concerned with first order Markov Chains, which satisfy the so-called *Markov Property*.

Definition 2. *The **Markov Property***

$$P[q_t = S_i \mid q_{t-1} = S_j] = P[q_t = S_i \mid q_{t-1} = S_j, ..., q_{t-k} = S_x]$$

states that the probability for state transition is solely dependent on the previous state, regardless of any other k states visited before that. Here, $q_t, q_t - 1$ and $q_t - k$ denote the chain's state at time step t, where S_i, S_j and S_x are sample states of the underlying system.

2.2. From Markov Chain Model to Hidden Markov Model

In some cases, the states in our Markov Chain Model are directly observable. However, suppose that the states we would like to reason about are not directly observable and are hidden. This motivates the extension of a Markov Chain Model to a *Hidden Markov Model*, in which the actual states we would like to reason about are hidden and cannot be observed. We can only observe *emission-signals*, which are being emitted from the hidden states. We follow Rabiners definition of HMMs, see [32].

Definition 3. *A **Hidden Markov Model** $\lambda = (\mathcal{A}, \mathcal{B}, \pi)$ with the hidden states $S = \{S_1, ..., S_N\}$ and emission signals $V = \{V_1, ..., V_M\}$ is a stochastic process, defined by three model parameters:*

1. *$N \times N$ row-stochastic **state transition matrix** $\mathcal{A}$, where the element a_{ij} denotes the state transition probability from state S_i to state S_j.*
   An alternative notation for denoting the state transition from q_{t-1} to q_t is given by $a_{q_{t-1}q_t}$.
2. *$N \times M$ row-stochastic **state emission matrix** $\mathcal{B}$ where the element b_{ij} denotes the probability $P[o_t = V_j | q_t = S_i]$ of observing emission signal $o_t = V_j$ at time step t under the hidden state S_i.*
   An alternative notation for denoting the observation of signal o_t from the hidden state q_t is given by $b_{q_t}(o_t)$.
3. *$1 \times N$ **initial state distribution vector** π, where $\pi_i = P[q_1 = S_i]$.*
   An alternative notation for denoting the initial state probability for hidden state q_t is given by π_{q_t}.

Situations where one needs to infer a sequence of hidden states from an observation sequence are very common. In the medical context, for example, we might be interested in inferring a sequence of diagnoses from the patients observed heartbeat and breathing frequency at certain time points.

2.3. Three Basic Problems for HMMs

Generally, we are concerned with inferring a sequence of hidden states from a sequence of observed states, a so-called *sequence-to-sequence* prediction. The Hidden Markov Model excels at this task, although we will see that the path towards a stable prediction poses some challenges and conceals certain pitfalls one needs to avoid. Following Rabiner [32], we see three main problems that have to be solved to be able to apply HMMs in practice. We will discuss their solution within the next sections.

- **Problem 1**: Given an observation sequence $\mathcal{O} = o_1 o_2 ... o_T$, as well as a fully parameterized model $\lambda = (\mathcal{A}, \mathcal{B}, \pi)$, how do we calculate the probability $P[\mathcal{O}|\lambda]$ in an efficient manner?

- **Problem 2**: Given an observation sequence $\mathcal{O} = o_1 o_2 ... o_T$ as well as a fully parameterized model $\lambda = (\mathcal{A}, \mathcal{B}, \pi)$, how do we find the state sequence $\mathcal{Q} = q_1 q_2 ... q_T$ best explaining the seen observation?
- **Problem 3**: Given a model λ, how do we train the model, changing its parameters to maximize $P[\mathcal{O}|\lambda]$?

Solving the first problem enables us to compare different Hidden Markov Models. Given an observation sequence $\mathcal{O}$, we might prefer the model λ, which maximizes $P[\mathcal{O}|\lambda]$, the probability of observing the given sequence based on model parameters.

A solution to the second problem allows for an inference of a sequence of hidden states $\mathcal{Q}$ given an observation sequence $\mathcal{O}$. Since many different hidden state sequences might produce the given observation sequence, the problem becomes finding a $\mathcal{Q}$, which maximizes $P[\mathcal{O}|\mathcal{Q}, \lambda]$, the probability of the hidden state sequence $\mathcal{Q}$ emitting the observation sequence $\mathcal{O}$.

The third problem is to approximate the in practice *unknown* model parameters $\mathcal{A}, \mathcal{B}, \pi$ given an observation sequence $\mathcal{O}$. This can be interpreted as training an HMM on an observation sequence.

2.3.1. Solution to Problem 1

Problem 1 is the problem of evaluating the probability of observing an observation sequence $\mathcal{O}$ given a model λ. Solving this problem enables us to compare different models, thus letting us decide which model "best fits" our observation. This problem is best approached naively at first, without taking computational costs into account. As we will see, the need for a smarter, less computationally intensive solution will arise along the way.

First, consider a fixed state sequence $Q = q_1 ... q_T$ of some states that might emit the observation sequence. The probability of this sequence arising from the model is given by the product of the probability of starting in the state q_1, which is given by π_{q_1} and the correct state transition probabilities.

$$P[Q|\lambda] = \pi_{q_1} \prod_{t=1}^{T-1} a_{q_t q_{t+1}} = \pi_{q_1} \cdot a_{q_1 q_2} ... \cdot a_{q_{T-1} q_T}.$$

Additionally, the probability of observing the observation sequence $\mathcal{O}$ from the state sequence Q is given by the product of the single emission probabilities for the individual observations under the hidden state.

$$P[\mathcal{O}|Q, \lambda] = \prod_{t=1}^{T} b_{q_t}(o_t) = b_{q_1}(o_1) \cdot ... \cdot b_{q_T}(o_T)$$

Finally, the probability for observing $\mathcal{O}$ under Q is $P[Q|\lambda] \cdot P[\mathcal{O}|Q, \lambda]$. Having computed this probability for one possible state sequence, all that is left is computing it again for every single possible state sequence of length T.

$$P[\mathcal{O}|\lambda] = \sum_{Q \in \mathcal{Q}} P[Q|\lambda] \cdot P[\mathcal{O}|Q, \lambda]$$

where, $\mathcal{Q}$ is the set of all possible state sequences of length T. Although very declarative, this solution is practically useless since the computational effort required to solve for only one observation sequence increases exponentially with the length T of the sequence, as for every time step there are up to N different state transitions to be made, resulting in a total of at worst N^T different candidates for Q, which are to be evaluated. This is unfeasible for even a small number of hidden states N and a short sequence length.

To overcome this hurdle, we make use of dynamic programming and temporarily store intermediate results to bootstrap and extend for a new, more optimal iteration of an intermediate result until we have reached the desired solution.

Definition 4. *The **forward variable** $\alpha_t(i) = P[o_1...o_t|q_t = S_i, \lambda]$ is defined as the probability of observing the partial observation sequence $o_1...o_t$ given State S_i at time step t as well as a fully parameterized model λ.*

We will use dynamic programming to compute the forward variable $\alpha_t(i)$ from our solutions for $\alpha_{t-1}(j)$, where $1 \leq j \leq N$ (see Figure 3). The full solution is as follows:

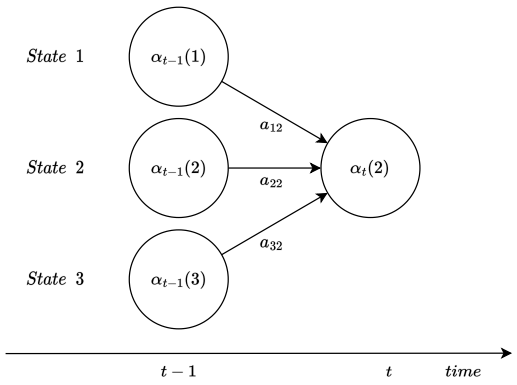

Figure 3. Usage of dynamic programming in the forward algorithm.

1. Initialization: $\alpha_1(i) = \pi_i \cdot b_i(o_1) \qquad 1 \leq i \leq N$
2. Induction:

$$\alpha_t(j) = \left(\sum_{j=1}^{N} \alpha_{t-1}(j) a_{ij} \right) \cdot b_j(o_{t-1}) \qquad 1 \leq j \leq N$$

$$1 < t \leq T$$

3. Termination: $P[\mathcal{O}|\lambda] = \sum_{i=1}^{N} \alpha_T(i)$

In the first step, we initialize the storage for the intermediate results, before continuously calculating the next intermediate results (see Figure 4). In the end, we sum over $\alpha_T(i)$ to obtain our desired result. This method of bootstrapping and reusing old results dramatically reduces the number of required operations down to N^2T (from the previous N^T) operations [32]. It should be noted that computing $\alpha_t(i)$ for every t and i for a given observation sequence $\mathcal{O}$ yields the so-called posterior distribution for the hidden states given the observation sequence. This will be important later on when we will discuss the prediction capabilities.

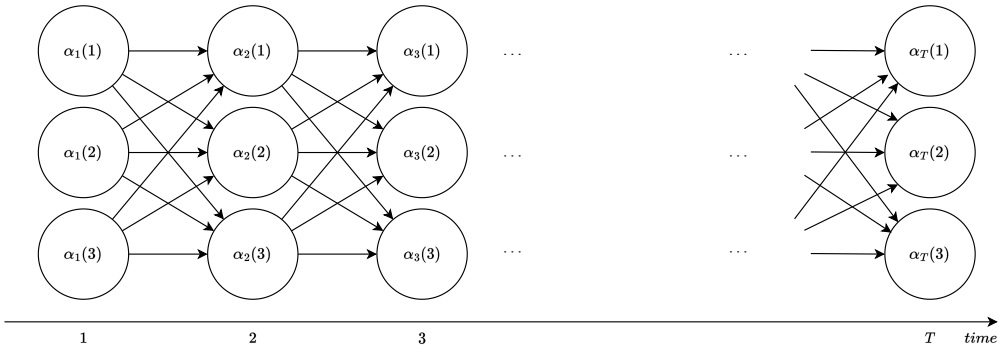

Figure 4. Visual representation of the full lattice structure used to compute $\alpha_T(i)$, with $1 \leq i \leq N$.

2.3.2. Solution to Problem 2

One possible solution to the question "*What is the most likely state-sequence for observation $\mathcal{O}$?*" is to find the most probable state S_i for each time step t. To tackle this problem, we

will need further definitions; first of all, let us define the so-called "backward-variable", which in its definition is quite similar to the forward variable.

Definition 5. *The **backward variable** $\beta_t(i) = P[o_{t+1}...o_T|q_t = S_i, \lambda]$ is defined as the probability of observing the partial observation sequence $o_{t+1}...o_T$ given State S_i at time step t as well as a fully parameterized model λ.*

To calculate the backward variable, we use the same idea of dynamic programming, reusing former results (see Figure 5). Although this time, we travel "backwards" through the observation sequence, thus starting at the last observation o_T.

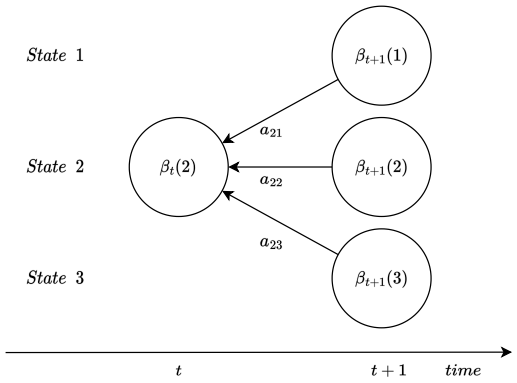

Figure 5. Usage of dynamic programming for the calculation of the backward variables.

1. Initialization: $\beta_T(i) = 1 \qquad 1 \leq i \leq N$
2. Induction:

$$\beta_t(i) = \sum_{j=1}^{N} \beta_{t+1}(j) a_{ij} b_j(o_{t+1}) \qquad 1 \leq i \leq N$$

$$1 \leq t < T$$

Having both, the forward and the backward variable at our disposal, we can continue defining a helper variable $\gamma_t(i)$.

Definition 6. *The **helper variable** $\gamma_t(i) = P[q_t = S_i|\mathcal{O}, \lambda]$ denotes the probability of being in the hidden state S_i at time step t given the full observation sequence $\mathcal{O}$ as well as a fully parameterized model λ.*

We can express the variable $\gamma_t(i)$ in terms of the forward and backward variables (see Figure 6): $\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P[\mathcal{O}|\lambda]} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}$. In this equation $P[\mathcal{O}|\lambda]$ is a normalization factor. Thus, $\sum_{i=1}^{N} \gamma_t(i) = 1$. To continue with our interpretation of optimality for a state sequence, we can solve for the individually most likely states for each time step: $q_t = \arg\max_{1 \leq i \leq N}(\gamma_t(i))$, $1 \leq t \leq T$. Unfortunately, there is a flaw in this solution. Although we have found the individually most likely states, we have no guarantee for soundness of the found sequence of states. It might just be that our sequence contains an illegal state transition. This error is resolved by the *Viterbi Algorithm* [33]:

Definition 7. *The **Viterbi Algorithm** finds the most likely state sequence $Q = \{q_1, q_2, ..., q_T\}$ to have emitted a given observation sequence $\mathcal{O} = \{o_1, o_2, ..., o_t\}$, whilst respecting the state transition constraints given by the model λ.*

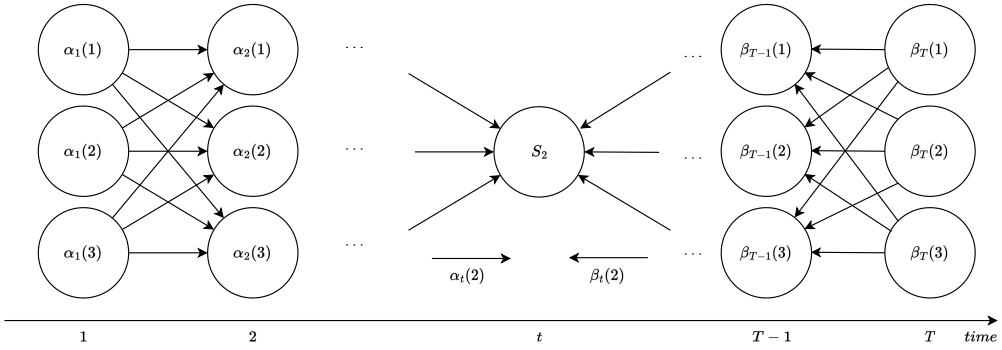

Figure 6. Usage of lattice structure in the forward-backward algorithm.

Definition 8. $\delta_t(i) = \max_{q_1,\ldots,q_{t-1}} P[q_1, ..., q_t = i, o_1, ..., o_t|\lambda]$ *is the probability for the most probable state sequence along a single path* $q_1, ..., q_t$ *with observation sequence* $o_1, ..., o_t$ *up until time step* t *that ends in the hidden state* S_i.

Again, we can compute this variable inductively by $\delta_t(j) = (\max_{1\leq i\leq N} \delta_{t-1}(i)a_{ij}) \cdot b_j(o_t)$. Since our goal is to find the optimal state sequence, we need a way to keep track of which prior state maximized $\delta_t(i)$ at each time step t for each state i. This is completed with a storage array $\psi_t(i)$. This lets us define the Viterbi Algorithm in four steps.

1. Initialization:

$$\begin{aligned} &\delta_1(i) = \pi_i b_i(o_1), \qquad 1 \leq i \leq N \\ &\psi_1(i) = 0 \end{aligned}$$

2. Recursion:

$$\begin{aligned} &\delta_t(j) = \max_{1\leq i\leq N}(\delta_{t-1}(i)a_{ij})b_j(o_t) && 1 \leq j \leq N \\ & && 2 \leq t \leq T \\ &\psi_t(j) = \arg\max_{1\leq i\leq N}(\delta_{t-1}(i)a_{ij}) && 1 \leq j \leq N \\ & && 2 \leq t \leq T \end{aligned}$$

3. Termination:

$$\begin{aligned} P^* &= \max_{1\leq i\leq N}(\delta_T(i)) \\ q_T^* &= \arg\max_{1\leq i\leq N}(\delta_T(i)) \end{aligned}$$

4. Backtracking: $q_t^* = \psi_{t+1}(q_{t+1}^*), \quad t = T-1, T-2, ..., 1$

2.3.3. Solution to Problem 3

Problem 3, which is concerned with training an HMM in a way such that the model is more likely to explain a given observation sequence, is the most difficult one out of the three presented problems. Starting from a rough estimate of the model parameters $\mathcal{A}, \mathcal{B}, \pi$, we can iteratively improve the model with the Baum–Welch Algorithm, which can be interpreted as an application of the Expectation-Maximization Algorithm to HMMs [34]; see also [32] for more details.

2.4. Machine Learning Metrics

Metrics are an important tool in ML to quantify the success of a system for a given dataset. They are a measure of quality for the learned model. Additionally, metrics allow the

comparison of different models with different architectures. The metric used is dependent on the type of prediction that is to be made, and since the following classification problem presented is of a multi-class nature, we will motivate and explain one suitable metric for these classification problems—the multi-class F_1-Score.

Definition 9. *We call* $F_\beta = (1+\beta^2) \cdot \frac{precision \cdot recall}{(\beta^2 \cdot precision)+recall}$ *the* F_β*-Score. For* $\beta = 1$*, we obtain the* F_1*-Score.*

The F_1-Score is a common metric used to measure the quality of binary classifiers. It represents the harmonic mean between precision and recall, two values we can compute with the help of C, our confusion matrix. $precision = \frac{true\ \ positives}{true\ \ positives + false\ \ positives}$, and $recall = \frac{true\ \ positives}{true\ \ positives + false\ \ negatives}$. We can extend the F_1-Score to the multi-class domain, by computing a $K \times K$ confusion matrix, where K is the number of classes our model can predict. Thus, we can compute the F_1-Scores for each class. The final F_1-Score is either an average or a weighted sum of the respective F_1-Scores [35].

3. Method

In this section, we will introduce the notation that will be used, and give an in-depth model overview.

3.1. Notation

Definition 10. *A* ***marker*** *M is a unique identifier or class-name for a certain set of discrete states* $S = \{S_1, ..., S_N\}$*. We say that* $\mathcal{M}$ *is the set of all markers.*

Markers are the abstract handle for the observable units in our environment. For example, whilst observing a medical patient, the set of markers $\mathcal{M}$ could for example include "ability to walk", "blood pressure", "breathing frequency". The set of concrete states of the marker "blood pressure" could be $S = \{low, medium, high\}$. $\mathcal{M}$ is partitioned into a set of layers $\mathcal{L}$.

Definition 11. *A* ***(prediction-) layer*** *L is an element of* $\mathcal{L}$*, partition of* $\mathcal{M}$*. A mapping* $\mathfrak{L}_{M \to L} : \mathcal{M} \to \mathcal{L}$ *maps a given marker M to its corresponding layer L.*

Using this definition, we can partition the set of markers into layers containing related markers. For example, we could group the markers "ability to walk" and "number of push-ups" into a layer "mobility". Next up, we should define the importance of these layers and their markers for our expected prediction. This is carried out by defining the following weight functions.

Definition 12. *Let a* ***layer-weight function*** $\omega_\mathcal{L} : \mathcal{L} \to \mathbb{R}_{\geq 0}$ *be a mapping from a layer L to a positive weight. The function* $\omega_\mathcal{L}$ *must satisfy* $\sum_{L \in \mathcal{L}} \omega_\mathcal{L}(L) = 1$*.*

Definition 13. *Let a* ***marker-weight function*** $\omega_L : L \to \mathbb{R}_{\geq 0}$ *be a mapping from a marker M of layer L to a positive weight. The function* ω_M *must satisfy* $\sum_{M \in L} \omega_L(M) = 1$*.*

Definition 14. *A* ***Trail*** $T = o_1 ... o_t$ *is an observation sequence of consecutive states of length t, which belong to a marker M. The weight of a trail is defined as* $\omega_T(T) = \omega_L(M) \cdot \omega_\mathcal{L}(L_{M \to L}(M))$*.*

Definition 15. *An* ***Observation*** $\mathcal{O} = \{T_1, ..., T_k\}$ *is a set of Trails of same length t. The mapping* $\mathfrak{M}_{T \to M} : \mathcal{O} \to \mathcal{M}$ *maps a Trail T to its corresponding marker M.*

Definition 16. *A* ***Query*** $\mathcal{Q} = (M_\mathcal{H}, \mathcal{L}, \mathcal{O}, \{\omega_{L_1}, ..., \omega_{L_k}\}, \omega_\mathcal{L})$ *consists of a hidden marker* $M_\mathcal{H}$ *as well as a partition of* $\mathcal{M}$*, namely* $\mathcal{L}$*. A Query possesses an observation* $\mathcal{O}$*, consisting of possibly*

many trails T. The weights of the markers and layers are defined by the marker-weight functions ω_{L_i} (one for each layer) and the layer-weight function $\omega_{\mathcal{L}}$.

Thus, a query $\mathcal{Q}$ is a well-defined description of a multivariate sequence alongside mixture components, namely the weights of the markers and layers. The mixture components are the weights, by which the individual result of each HMM is weighted. We would like to continue to define a formalism, which maps augmented versions of these queries to a prediction result. Thereby, the form of the prediction is dependent on the augmented query. Plainly said, our model can predict time series of discrete states, as well as time series of distributions over states. At first, we will keep the prediction result—in any case, some sort of time series—very abstract.

Definition 17. *A **trail-evaluation** $\phi(T, M_{\mathcal{H}}) = \omega_T(T) \cdot P(T, \lambda(M(T), M_{\mathcal{H}}))$ is a function that maps a Trail, a Query and possibly various arguments to a specifically requested weighted prediction result $P(T, \lambda(M(T), M_{\mathcal{H}}))$.*

Here, $\lambda(M(T), M_{\mathcal{H}})$ is a function that returns a fully parameterized HMM whose hidden states are the states of the hidden marker $M_{\mathcal{H}}$ and the observable states are the states of the trail marker $M(T)$. Subsequently, the resulting HMM is used to reason about the given trail T. Further details about the nature of the prediction are given in Section 3.3.4.

Definition 18. *An **observation–evaluation** $\Phi(\mathcal{Q}) = \sum_{T \in \mathcal{Q}} \phi(T, M_{\mathcal{H}}), \quad M_{\mathcal{H}} \in \mathcal{Q}$ is a function that sums up the weighted prediction results to construct the final prediction.*

Hence, an observation–evaluation of a query is the weighted sum of the evaluations of the trails—the constituents of the observation.

3.2. Model Overview

Observing an environment in the real-world allows for the observation of the states of multiple markers M_i at each time step. Certain semantically related markers are grouped into layers. We would like to reason about a hidden state in this environment based upon our observations by constructing a query $\mathcal{Q}$. The idea is to construct an HMM for every single marker and try to infer knowledge about the hidden states from given observations, according to the weight of the marker $\omega_L(M)$ and the weight of its respective layer $\omega_{\mathcal{L}}(L(M))$. The resulting model can briefly be described as a mixture of Hidden Markov Models—one HMM for each marker. It is important to understand that our model is only able to offer prediction capabilities that a simple one-observation HMM could offer as well.

In this work, we assume the hidden marker to be one of the visible observation markers, allowing for the ability to generically switch the hidden marker and prediction-layer to allow for maximum flexibility. Hence, we will be able to extract the parameters of the model, namely $\mathcal{A}, \mathcal{B}$, and π from the observation sequences directly, instead of having to train the model based on said sequences. This is in stark contrast to the usual usage of HMMs, where these parameters are not given and have to be approximated (learned) from observations.

3.3. Pipeline Description

The prediction pipeline was written in Python, making use of the `hmmlearn` library (the pipeline and documentation is available at https://github.com/rfechner/generic-hmm (accessed on 26 March 2024)). The pipeline comprises a pre-processing step, a feature extraction step, and a prediction step (see Figure 7). To obtain a trained model, the user must input their training data in the form of a csv-file as well as an ini-config file, describing the layer structure and provide sufficient information about the weights of the model.

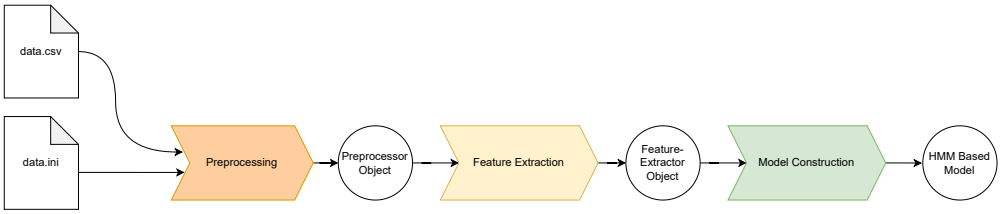

Figure 7. Flow-diagram of the prediction pipeline. The data is converted to a fully functioning model.

3.3.1. User Input

The data consisting of the different observations provided by the csv-file, as well as the configuration of the model provided by the ini-file, must be supplied by the user. The abstract syntax outlines the correct way of specifying an ini-file required to construct a model, whilst making no assumption about the form of the data; thus, it uses the Backus Naur Form (BNF, see [36]). Each marker in the data must be specified as a section inside the file. Additionally, information about the datatype, related layer, and layer-specific weight of the marker must be supplied as a key-value pair under the corresponding markers section. Optional information, like the relationship to other markers, can be added.

It should be noted that, although the definition of a marker calls for the existence of a layer-specific weight, the program is robust against missing or faulty weights and will re-balance the given weights to satisfy stochastic constraints.

3.3.2. Pre-Processing

Pre-processing is a modular stage inside the pipeline, which itself is a small pipeline (see Figure 8). The transformation of the data supplied includes the analysis of the ini-file, conditional deletion of any unnecessary data, the grouping of the data according to the metainformation extracted from the ini-file, enforcing measurement interval consistency if necessary, and finally encoding the data into a less memory intensive format.

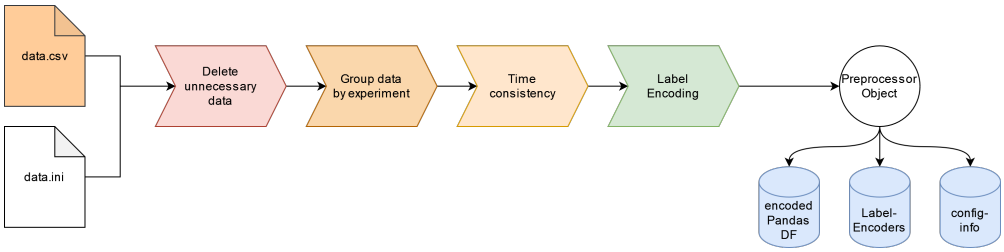

Figure 8. Flow-diagram of the pre-processing pipeline.

3.3.3. Feature Extraction

The feature extraction builds upon the previous step in the pipeline, the pre-processing (see Figure 9). Just as the prior component of the pipeline, the feature extraction component is fully modularized, implementing the necessary interface used to provide the required functionality to the next part in the pipeline. Inside the feature extraction stage, the *state transition-*, *signal emission-* and *initial state*-probabilities are extracted from the encoded data supplied by the pre-processing stage. This is an important step, since we can use the extracted probabilities later on to construct HMMs with a strong initial guess for $\mathcal{A}, \mathcal{B}$ and π.

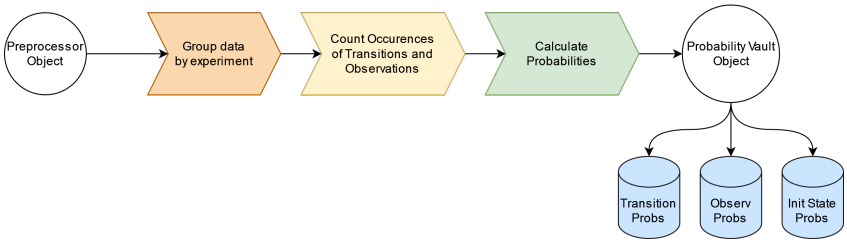

Figure 9. A flow-diagram of the Feature Extraction Pipeline.

3.3.4. Model Validation and Query

Finally, in the last step of the pipeline, the HMM-based model is constructed and queried. Building on top of the previously constructed feature extraction, the actual construction of the different HMMs is very convenient. It includes the interpolation of the results given by the individual HMMs according to the weights specified inside the configuration. To give a measure of success, the model is able to compute the multi-class F_1-Score for a given validation dataset.

As already stated, it is important that the HMM-based model can only offer predictions that a simple single observation HMM could offer as well. These predictions include the following:

1. **Posteriors for each hidden state for observation $\mathcal{O}$**
   Given a Query $\mathcal{Q}$, compute the posterior distribution for each hidden state of $M_{\mathcal{H}}$ for every time step t given the trails T_i. The result will be a weighted sum (according to the weights defined in $\mathcal{Q}$) of the individually computed posteriors.
2. **Distribution over hidden states following $\mathcal{O}$**
   Given a Query $\mathcal{Q}$, predict the distribution over the hidden states of $M_{\mathcal{H}}$ for possibly many time steps $\hat{t}$ following the observation. This yields an approximation to a stationary distribution of the state transition matrix for $M_{\mathcal{H}}$. The kind of stationary distribution is dependent on the initial state distribution given by the observation sequence $\mathcal{O}$.
3. **Optimal state sequence**
   Given a Query $\mathcal{Q}$, compute the optimal state sequence of $M_{\mathcal{H}}$ "best explaining" the trails T_i using the Viterbi Algorithm.

3.3.5. Controller

To wrap all of these components up, and use the whole pipeline, as well as enable plotting of the results, a wrapper object called `Controller` provides a user-friendly interface.

3.3.6. Hyperparameter Optimization

An advantage of including more hyperparameters like the marker weights ω_{L_i} is that we are able to include more prior domain knowledge into a prediction, the flipside being that we do have to provide said domain knowledge in the first place, creating a dependence on domain experts. A possible solution would be to assign equal weights to all markers, however in some cases we would like to automatically tune these hyper-parameters.

This is possible by numerically optimizing the data-likelihood $P[\mathcal{O}|\lambda; \omega_{L_1}, \ldots, \omega_{L_k}]$ with respect to the layer weights $\theta := \omega_{L_i}$. The data likelihood can be computed using the *forward*-algorithm, as $P[\mathcal{O}|\lambda; \theta] = \sum_i^N \alpha_T(i)$. Importantly, we have to make sure that we stay within the stochastic constraints, namely that the marker weights are a partition of the probability mass assigned to the respective layer. We used the differentiable softmax (the softmax maps a vector of reals to a valid categorical distribution) function, allowing the optimizer to operate on mappings of the parameters in the reals, while respecting the optimization constraints. More specifically, every layer L is assigned probability mass m_L. The subset of hyper-parameters belonging to layer L are piped through a softmax, before being normalized with the respective weight m_L. This ensures that the total probability mass assigned to the markers (i.e., the individual HMMs) sums to 1. We denote the layer-seperate application of the softmax and re-scaling as lws (layer-wise softmax). The found hyper-parameters can then be extracted by applying the layer-wise softmax to the found local minimum. In our experiments, we augmented the objective with entropy regularization, pushing the solution towards a more uniform distribution. We used a constant stepsize α for the regularization objective. The corresponding loss function is given as:

$$\mathcal{L}(\theta) = \underbrace{-\log P[\mathcal{O}|\lambda; \mathrm{lws}(\theta)]}_{\text{NLL}} + \underbrace{\alpha \mathcal{H}(\mathrm{lws}(\theta))}_{\text{Entropy reg.}}$$

We used an off-the-shelf quasi-Newton optimizer (https://docs.scipy.org/doc/scipy/reference/optimize.minimize-bfgs.html#optimize-minimize-bfgs (accessed on 26 March 2024)) [37] with standard parameters for the optimization.

3.4. Data

3.4.1. Data Generation

The random dataset mimics the actual biomedical dataset in which we expect to see many markers with degenerative states—markers, whose states continuously progress over time, going from an initial good state into a worse state. Four different degenerative markers were selected, namely

1. $M_{finemotor}$, a marker whose state indicates the subject's ability to solve tasks using their hands.
2. $M_{mobility}$, a marker whose state indicates the subject's state of mobility, e.g., still being able to walk freely without the need of walking aids or the need to use a wheelchair.
3. $M_{neuropsych}$, a marker whose state indicates the subject's ability to solve mental tasks.
4. $M_{diagnosis}$, a marker whose state indicates the diagnosis given by an expert for a subject at a certain time step.

Every marker has the same set of degenerative states, namely $S = \{\text{good, med-good, med, med-bad, bad, severe}\}$ indicating the current state under the given marker. To construct the data, a state transition matrix $\mathcal{A}$ as well as an initial state distribution vector π were constructed for the single marker $M_{diagnosis}$. Additionally, emission signal matrices ($\mathcal{B}_{motoric}$, $\mathcal{B}_{mobility}$, $\mathcal{B}_{neuro}$) were constructed for the other markers $M_{motoric}$, $M_{mobility}$, M_{neuro}. Using these parameters, three different HMMs were constructed. Using the initial state probability distribution, for each observation sequence, an initial state for the marker $M_{diagnosis}$ was chosen. Thereafter, the following states for $M_{diagnosis}$ were sampled from the state transition probability matrix $\mathcal{A}$. Likewise, for each time step of each observation, the emission signal for each observation marker M_i was sampled from the distribution located in the respective emission signal matrix $\mathcal{B}_i$. A dataset of $N = 300$ observation sequences of variable sequence-length and variable time intervals in between measurements was synthesized. To finalize the data, a configuration ini-file was constructed. For simplicity, each marker was assigned to a distinct layer, resulting in four layers—one for each marker.

3.4.2. Background and Description of the Biomedical Dataset

Here, we will discuss a dataset obtained by a longitudinal observational study of subjects suffering from spinocerebellar ataxia type 3 (SCA3), the most common autosomal dominantly inherited neurodegenerative ataxia disorder (ESMI, European spinocerebellar ataxia type 3/Machado–Joseph disease initiative [9–12]). Symptoms include progressive loss of balance, coordination deficits and slurred speech. SCA3 patients experience significant restrictions of mobility and communicative skills. Preventive interventions that aim to silence the disease gene offer a promising treatment option. The first clinical gene silencing trial (ClinicalTrials.gov (accessed on accessed on 26 March 2024), Identifier: NCT05160558) has recently started. Consequently, there is an urgent need to predict the deterioration to a more severe disease stage to prioritize and treat patients suffering from a greater risk with less uncertainty.

The supplied data contain results from various scales that were assessed in all participants on an almost annual basis. Usually, those scales reflect increased impairment with higher scoring, while 0 refers to a normal condition and the absence of signs or symptoms like in the healthy population. In our analyses, we included the following 4 assessments that measure special ends of abilities or measured observations of subjects, where a rating of 0 indicates a healthy condition: (1) Disease staging (ranging from 0 = normal over 1 = ataxic,

but able to walk freely, 2 = need to use walking aids, 3 = requirement to use wheelchair) [38]; (2) Scale for the assessment and rating of ataxia (SARA), measuring the ataxia severity rated in 8 different items (namely gait, stance, sitting, speech, finger chase, nose-finger test, fast alternating hand movements and heel-shin slide) resulting in a sum score (raging from 0 to 40) [39]; (3) Inventory of non-ataxia signs (INAS), assessing neurological symptoms other than ataxia [40]; and (4) A scale assessing the activities of daily life (ADL) [41]. Notably, the "ADL" (Activities of Daily Life), "INAS" (Intentional Non-Adherence Scale) and "SARA" (Scale for the Assessment and Rating of Ataxia) scales were considered in the first part of the experiment. These specific scales were chosen after consulting with a domain expert. With the additional information about the disease state—already inferred by domain experts—for each multivariate observation in our data, we return to the familiar setup presented in the prior section, where we have multiple trails paired with a given diagnosis or inferred latent state. $M_{diagnosis}$ in this context represents the disease stage (normal, ataxic, walking aids, wheelchair). A "diagnosis" in its native meaning, i.e., usually describing a particular disease, does not make sense in the context of a hereditary disease that is "diagnosed" simply by a genetic test. Thus, instead of the 3 observation markers like in the prior example, we here obtain about 109 different markers, which we can observe at every time step. The average length of the given time-series was calculated to be about 2.3 time steps. Thirty-one subjects were excluded due to missing or invalid genetic information, sixteen subjects were excluded due to invalid SARA item scores (×0.5-scoring in items that only allow for integer values, namely gait, stance, sitting, and speech) and five subjects for which only characterizing and demographic but no visit-specific data were available. The data was grouped according to the unique patient identifier in order to assign all visits to this particular participant.

4. Results

The model was evaluated on two different datasets: a synthetic dataset (published with this paper) and a real-world biomedical dataset containing observational data of a longitudinal natural history study in spinocerebellar ataxia type 3 (SCA3). Due to data-protection reasons, we will present an in-depth discussion of results and evaluation for the synthetic dataset, which was conceived to mimic the real-world dataset.

4.1. Evaluation 1: Random Data

The dataset was evaluated with a 10-fold cross validation. The layers of query $\mathcal{Q}$ were set to $\mathcal{L} = \{ L_1 = \text{mobility}, L_2 = \text{motoric}, L_3 = \text{neuro}\}$. In this special case, every marker has an equal influence on the model prediction, since every layer consists of only one marker and the layer weights are equally distributed across all layers. Alternatively, the layer weights $w(L_i)$ could be adjusted, or multiple markers could be grouped in one layer, allowing for further differentiation by adjusting the marker weights $w(M_i)$ inside one layer. The hidden marker $M_{\mathcal{H}}$ was set to the marker $M_{diagnosis}$.

As already stated above, the model was evaluated with a 10-fold cross validation, which yielded an average $\boldsymbol{F_1}$-Score of 0.81 ± 0.02, Recall of 0.81 ± 0.02 and Precision of 0.82 ± 0.02. To showcase the prediction capabilities of the model further, a never-seen-before observation was generated, and the model was queried for all three possible prediction types (see Section 3.3.4). These include the prediction of the distribution over the posteriors and an extrapolation for the posterior distributions given an observation sequence (see Figure 10). A thourough analysis including classification reports and confusion matrices is given in the notebooks rendered in our GitHub repository (https://github.com/rfechner/generic-hmm/blob/main/tutorial.ipynb (accessed on 26 March 2024)).

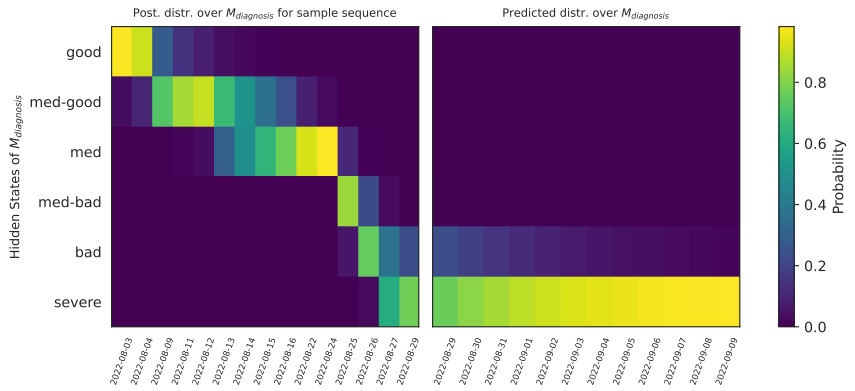

Figure 10. The posterior distribution over $M_{\mathcal{H}} = M_{\text{diagnosis}}$ (**left**). Every column is a categorical distribution over the hidden states. We can see that the estimated probability for the "good" state diminishes over time, whereas the estimated probability for the "severe" state increases. Extrapolated distribution (**right**) under $M_{\mathcal{H}} = M_{diagnosis}$ changes over time. We unroll the hidden Markov Chain to predict steps into the future.

4.2. Evaluation 2: Biomedical Data

4.2.1. First Experiment

The 109 markers were partitioned into three layers corresponding to the scale each marker belongs to; L_{ADL}, L_{INAS} and L_{SARA}. The weight function ω_L assigns each prediction-layer the weight $\frac{1}{3}$. Internally, the weight function responsible for assigning a weight to every marker inside a given layer assigns equal weight to every marker. The hidden marker $M_{\mathcal{H}}$ was chosen to be the diagnosis marker. This proved to be a naive approach, as the bad performance of the model with an F_1-Score of less than 0.5 in a 10-fold cross-validation of the whole dataset reassured.

4.2.2. Second Experiment

Since first experiments yielded an unacceptable performance, the decision was made to reduce the number of Trails, by only considering the cumulative score for each of the three scales (ADL, INAS and SARA), drastically reducing the number of Trails down to only three. Within this reduced setting, the model performed relatively well, with an F_1-Score of 0.75 ± 0.04, Recall of 0.78 ± 0.05 and Precision of 0.76 ± 0.06 as the result of a 10-fold cross validation on the whole dataset (note that metrics were computed as macro-averages over multiple classes). Additionally, further constellations of prediction-layers and hidden markers were tested (see Table 1). The predictions with $M_{\mathcal{H}}$ ="diagnosis" generally produced acceptable results, whereas the other predictions yielded rather unsatisfying prediction results.

Table 1. Table cells display the mean and standard deviation of the F_1-Scores received from a 10-fold cross validation. Different constellations of prediction-layers and hidden marker were used. Overall on the level of single layers LSARA allows the highest prediction of disease staging $M_{diagnosis}$ (normal, ataxic, walking aid, wheelchair), followed by LADL, representing the activities of daily living, while neurological symptoms other than ataxia (LINAS) do not allow any meaningful prediction.

F_1-Scores for 10-Fold Cross Validation ($\mu \pm \sigma$)				
Layers \ $M_{\mathcal{H}}$	**Diagnosis**	**ADL Score**	**INAS Score**	**SARA Score**
$L_{ADL}, L_{INAS}, L_{SARA}$	0.75 ± 0.04	0.22 ± 0.06	0.13 ± 0.04	0.19 ± 0.06
L_{SARA}	0.75 ± 0.03	0.23 ± 0.03	0.16 ± 0.04	-
L_{INAS}	0.6 ± 0.06	0.11 ± 0.04	-	0.13 ± 0.04
L_{ADL}	0.72 ± 0.08	-	0.19 ± 0.05	0.24 ± 0.04

5. Discussion

Generally, it was expected that the model would perform relatively well given the right prediction-layers and enough training data. A substantial difficulty with the real-world data we had access to, was that a majority (approx. $\frac{2}{3}$) of the given training sequences was of length 1 or 2; thus, the dataset lacked sufficient samples for longer time-series. Naturally, this also affected the validation dataset. This leads to an important question: Which clinical trials or patient studies produce data suitable for an HMM-based prediction? It was shown in Section 4.2.2 that we can reach reasonable prediction scores using an HMM approach—at least for the right constellations of prediction-layers and hidden marker. To this end, we also showed that numerical optimization of the layer weights is possible. For limited test data, we were able to archive higher prediction scores, for some we did not. Here, some more work is to be conducted, which is why we reported the results of the vanilla non-hyperparameter optimized model. The model seems to generalize quite nicely across all of the validation sets; most importantly, the results we obtain from the k-fold cross validation are quite low in variance, which we attribute to the stability induced by the ensemble prediction of the HMMs.

Reasonable results were produced, given the prediction-layers L_{ADL}, L_{INAS}, L_{SARA} in constellation with the hidden marker $M_{\mathcal{H}}$ set to the marker "diagnosis" can be explained by the strong correlation between the individual markers inside the prediction-layers (which are simply the scores for the corresponding scale) and the diagnosis given by a medical professional. Specifically, the prediction-layer L_{SARA} seems to be the most suitable for producing predictions for the hidden marker "diagnosis". This independently received finding correlates with the findings of domain experts, i.e., biomedical experts.

We also need to discuss the limitations of our approach. First, as discussed above, the time-series as base for prediction, as well as the distribution for the initial hidden state have to be considered. As already described, about $\frac{2}{3}$ of the training data contain time-series only of length 1 or 2. This greatly influences the prediction capabilities for longer time-series. The dataset does not have enough training samples for longer time-series, thus the performance for predicting longer sequences could be better. Additionally, since we extract the probabilities from the training data, the model has a strong bias for the probability of the initial state. Without knowing anything about the validation sample, a certain initial state is far more probable than another. This fact, paired with the appearance of short (sometimes single time step) time-series, is most definitely a source of errors in our prediction.

Furthermore, an analysis of the individual time-series and the corresponding scales of the real-world data set in patients has shown that the underlying Markov Chains for the markers are not strictly left-right (left–right HMMs model strictly degrading states, meaning once a state transition from A to B has occurred, it is impossible to reverse this transition. A will never be reached again). To put it in other words, the difference in, e.g., the ADL score, a measure of the subject's ability to manage activities of daily living, between the last and first measurement is not strictly positive. In some cases, we can observe that a patient reduces their score over time, instead of displaying a degenerating performance. There are many potential reasons for such an unexpected improvement over time. One explanation can be an initially present comorbidity causing an impairment beyond the ataxia disorder, e.g., a broken arm. Another, probably more common, explanation might be a rater depended on variance. In the case of predicting the hidden markers "ADL Score", "INAS Score" or "SARA Score", the granularity of the discretization of the continuous scales deeply impacts the prediction score. For the given data, the granularity (i.e., the dtype linspace defined in the configuration) was kept the same for all measurements.

5.1. Strengths, Weaknesses and Improvements

The HMM-based model brings interpretability and explainability by nature. One may inspect the transition and emission probability matrices after training to gain insight on dynamics and or common confounds between variables. Additionally, multiple versions

of the same model trained on subsets of data may give insight into how beliefs about dynamics change on a meta-level. Since the model can be seen as a mixture of HMMs, its capabilities to capture deep complexity and intricate relationships within data are limited. As the model can use multiple markers for prediction, the general stability of a prediction is traded for a lack of attention to details in the data. Since we weigh the influence the markers have on the prediction separately, small but maybe very critical changes in a single marker might be overpowered by the sheer number of markers our model has to respect. The only way to combat this problem right now would be to assign a higher weight to critical markers, in order to let small changes in states for an important marker have a higher impact on the final prediction. However, a data-driven solution that includes an automated identification of helpful weightings would be desirable. This was realized by employing gradient descent on the negative data log likelihood with additional entropy regularization to obtain "smoother" solutions to the optimization objective. In this paper, we have trained HMMs in a supervised way, introducing information about the hidden states and the transitions between them into the model. Later on the model could be fine-tuned on new observation lacking information about the hidden state. This would be carried out using the EM-Algorithm, which is prone to overfitting. Additionally, for the ensemble of HMMs every constituent of the Ensemble has to be optimized seperately, as we make the naive Bayes assumption. Although there is not an embedded ranking of feature importances to HMMs, one may confounds, overlap and correlations in features by inspecting the emission matrices after training. Another limitation of HMMs is that they tend to overfit when extensively trained. The hyper-parameter optimization also seems to be quite unstable, finding good minima in only about 20% of the initializations.

5.2. Model Capabilities and Usecases

During the design of the pipeline, we paid special attention to usability. The prediction target and result can be altered by specifying the prediction-layers and hidden marker, but also by specifying the weight functions for the layers and their respective markers, as well as the partition of the markers into layers in the first place. Domain experts are free to adjust these parameters within the constraints of the proposed abstract syntax. Apart from configuring the model's parameters, domain experts are presented with multiple prediction capabilities:

1. *Posteriors for states of a hidden marker*: The prediction tool is able to predict and plot posteriors for the states of a hidden marker $M_{\mathcal{H}}$. For a given observation sequence, the model predicts $\alpha_t(i)$ for every state i and every time step t of the observation sequence. This enables the user to produce a simple visualization of how likely the model thinks a certain hidden state for the given time step of the observation. A domain expert can see at a glance in which direction the states of $M_{\mathcal{H}}$ evolve.
2. *Approximation of the stationary distribution*: Additionally, the model can give a rough estimate of the "future" posteriors of the states of $M_{\mathcal{H}}$ beyond the given observation sequence. In other words, it can give an estimate about how the distribution over the probability for the states of $M_{\mathcal{H}}$ will evolve over future time steps. It should be added that this estimate is a visualization of the approximation of the stationary distribution of the state transition matrix $\mathcal{A}$ of the underlying Hidden Markov Model.
3. *Optimal state sequence prediction*: The most valuable prediction capability might be the prediction of optimal state sequences given an observation sequence. This allows for "double-checking" already-found observation sequences, and might be used for data augmentation in the case of missing data or that a malfunctioning sensor gives back erroneous measurements. Predicting an optimal sequence of states for a hidden marker "diagnosis" might be of help to a medical professional, who wants to verify their given diagnosis.
4. *Extraction of model parameters*: Finally, a user can extract the model parameters themselves, as they can give us critical information about the general transition probabilities of a model. A domain expert might ask about a rough estimate of the probability

for state transitions, which they could immediately obtain by extracting the state transition probability matrix $\mathcal{A}$ from the model.

6. Conclusions

Biomedical data often contain longitudinal data, for example biomedical information on disease progress. An important goal is to infer the unknown solely from observation. Hidden Markov Models (HMMs) have been successfully applied to the processing of possibly noisy continuous signals. Here, we presented a novel approach based on multivariate time-series of categorically distributed data.

We provided a prediction pipeline system, which processes data paired with a configuration file, enabling us to construct, validate and query a fully parameterized HMM-based model. It has been conceptualized to be highly customizable and accessible both to computer scientists and practitioners from other disciplines, for example biomedical research. In addition, we provided a theoretical and practical framework for multivariate time-series inference based on HMMs that included constructing multiple HMMs to predict another observable variable. Our analysis was carried out on random data, but also on biomedical data based on Spinocerebellar ataxia type 3 disease.

The implementation of the HMM framework is publicly available and can be easily configured and adapted for further experiments. We could show that our proposed approach shows promising results when tested on real-world application data. However, we also presented a detailed discussion and how the approach could be augmented to improve results. It is obvious that the selected input data must meet certain criteria, especially with respect to the quantity of available longitudinal time points, to allow reasonable predictions. But even in a relatively small data set of rare diseases, we could demonstrate the feasibility of our approach. The presented framework has to be proven on further different biomedical time-series in order to fully estimate its potential and moreover how the suggested augmentations might improve the overall prediction.

Author Contributions: Conceptualization, R.F.; Data curation, R.F.; Formal analysis, R.F., J.D. and R.R.; Funding acquisition, R.R. and J.F.; Methodology, R.F., J.D. and R.R.; Project administration, J.D., R.R. and J.F.; Resources, J.D. and J.F.; Software, R.F.; Supervision, J.D. and R.R.; Validation, J.D., R.R. and J.F.; Visualization, R.F.; Writing—original draft, R.F. and J.D.; Writing—review and editing, J.D., R.R. and J.F. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by a postdoc fellowship of the German Academic Exchange Service (DAAD), granted to RR. JF receives funding of the National Ataxia Foundation (NAF) and as a fellow of the Hertie Network of Excellence in Clinical Neuroscience. This publication is an outcome of ESMI, an EU Joint Programme—Neurodegenerative Disease Research (JPND) project (see www.jpnd.eu (accessed on 26 March 2024)). The project is supported through the following funding organisations under the aegis of JPND: Germany, Federal Ministry of Education and Research (BMBF; funding codes 01ED1602A/B); Netherlands, The Netherlands Organisation for Health Research and Development; Portugal, Foundation for Science and Technology and Regional Fund for Science and Technology of the Azores; United Kingdom, Medical Research Council. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 643417. Furthermore, the publication was supported by the Open Access Fund of Universität Koblenz.

Institutional Review Board Statement: Included are retrospective data of previously published observational studies. All studies were conducted according to the guidelines of the declaration of Helsinki and approved by the ethics committees of contributing centers.

Informed Consent Statement: Informed written consent was obtained from all study participants.

Data Availability Statement: The pipeline and additional material is available at https://github.com/rfechner/generic-hmm accessed on 26 March 2024). The biomedical data that support the findings of this study underlie data protection policies and are therefore not publicly available. However, they can be made available upon reasonable request with permission of the ESMI consortium (contact: Jennifer Faber, jennifer.faber@dzne.de).

Acknowledgments: We thank Jan Jürjens, University of Koblenz, for his contributions to the initial research.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Cheng, Y.; Wang, F.; Zhang, P.; Hu, J. Risk prediction with electronic health records: A deepearning approach. In Proceedings of the 2016 SIAM International Conference on Data Mining, Miami, FL, USA, 5–7 May 2016; SIAM: Philadelphia, PA, USA, 2016; pp. 432–440.
2. Ferreira, M.I.A.; Barbieri, F.A.; Moreno, V.C.; Penedo, T.; Tavares, J.M.R. Machineearning models for Parkinson's disease detection and stage classification based on spatial-temporal gait parameters. *Gait Posture* **2022**, *98*, 49–55. [CrossRef] [PubMed]
3. Nash, C.; Nair, R.; Naqvi, S.M. Machineearning and ADHD mental health detection—A short survey. In Proceedings of the 2022 25th International Conference on Information Fusion (FUSION), Linköping, Sweden, 4–7 July 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1–8.
4. Placido, D.; Yuan, B.; Hjaltelin, J.X.; Zheng, C.; Haue, A.D.; Chmura, P.J.; Yuan, C.; Kim, J.; Umeton, R.; Antell, G.; et al. A deepearning algorithm to predict risk of pancreatic cancer from disease trajectories. *Nat. Med.* **2023**, *29*, 1113–1122. [CrossRef] [PubMed]
5. Mall, S.; Srivastava, A.; Mazumdar, B.D.; Mishra, M.; Bangare, S.L.; Deepak, A. Implementation of machineearning techniques for disease diagnosis. *Mater. Today Proc.* **2022**, *51*, 2198–2201. [CrossRef]
6. Liu, S.; Masurkar, A.V.; Rusinek, H.; Chen, J.; Zhang, B.; Zhu, W.; Fernandez-Granda, C.; Razavian, N. Generalizable deepearning model for early Alzheimer's disease detection from structural MRIs. *Sci. Rep.* **2022**, *12*, 17106. [CrossRef] [PubMed]
7. Adler, D.A.; Wang, F.; Mohr, D.C.; Choudhury, T. Machineearning for passive mental health symptom prediction: Generalization across differentongitudinal mobile sensing studies. *PLoS ONE* **2022**, *17*, e0266516. [CrossRef] [PubMed]
8. Barough, S.S.; Safavi-Naini, S.A.A.; Siavoshi, F.; Tamimi, A.; Ilkhani, S.; Akbari, S.; Ezzati, S.; Hatamabadi, H.; Pourhoseingholi, M.A. Generalizable machineearning approach for COVID-19 mortality risk prediction using on-admission clinical andaboratory features. *Sci. Rep.* **2023**, *13*, 2399. [CrossRef] [PubMed]
9. Faber, J.; Schaprian, T.; Berkan, K.; Reetz, K.; França, M.C., Jr.; de Rezende, T.J.R.; Hong, J.; Liao, W.; van de Warrenburg, B.; van Gaalen, J.; et al. Regional Brain and Spinal Cord Volume Loss in Spinocerebellar Ataxia Type 3. *Mov. Disord.* **2021**, *36*, 2273–2281. [CrossRef] [PubMed]
10. Wilke, C.; Haas, E.; Reetz, K.; Faber, J.; Garcia-Moreno, H.; Santana, M.M.; van de Warrenburg, B.; Hengel, H.; Lima, M.; Filla, A.; et al. Neurofilaments in spinocerebellar ataxia type 3: Blood biomarkers at the preataxic and ataxic stage in humans and mice. *EMBO Mol. Med.* **2020**, *12*, e11803. [CrossRef] [PubMed]
11. Garcia-Moreno, H.; Prudencio, M.; Thomas-Black, G.; Solanky, N.; Jansen-West, K.R.; Hanna Al-Shaikh, R.; Heslegrave, A.; Zetterberg, H.; Santana, M.M.; Pereira de Almeida, L.; et al. Tau and neurofilamentight-chain as fluid biomarkers in spinocerebellar ataxia type 3. *Eur. J. Neurol.* **2022**, *29*, 2439–2452. [CrossRef]
12. Hubener-Schmid, J.; Kuhlbrodt, K.; Peladan, J.; Faber, J.; Santana, M.M.; Hengel, H.; Jacobi, H.; Reetz, K.; Garcia-Moreno, H.; Raposo, M.; et al. Polyglutamine-Expanded Ataxin-3: A Target Engagement Marker for Spinocerebellar Ataxia Type 3 in Peripheral Blood. *Mov. Disord.* **2021**, *36*, 2675–2681. [CrossRef]
13. Ashizawa, T.; Öz, G.; Paulson, H.L. Spinocerebellar ataxias: Prospects and challenges for therapy development. *Nat. Rev. Neurol.* **2018**, *14*, 590–605. [CrossRef]
14. Klockgether, T.; Mariotti, C.; Paulson, H.L. Spinocerebellar ataxia. *Nat. Rev. Dis. Prim.* **2019**, *5*, 24. [CrossRef] [PubMed]
15. Baker, J. The DRAGON system–An overview. *IEEE Trans. Acoust. Speech Signal Process.* **1975**, *23*, 24–29. [CrossRef]
16. Nilsson, M.; Ejnarsson, M. Speech Recognition Using Hidden Markov Model. 2002. Available online: https://www.diva-portal.org/smash/get/diva2:831263/FULLTEXT01.pdf (accessed on 26 March 2024).
17. Lee, H.K.; Kim, J.H. An HMM-based threshold model approach for gesture recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **1999**, *21*, 961–973.
18. Frasconi, P.; Soda, G.; Vullo, A. Text categorization for multi-page documents: A hybrid naive Bayes HMM approach. In Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital Libraries, Roanoke, VA, USA, 24–28 June 2001; pp. 11–20.
19. Vairavan, S.; Eshelman, L.; Haider, S.; Flower, A.; Seiver, A. Prediction of mortality in an intensive care unit usingogistic regression and a hidden Markov model. In Proceedings of the 2012 Computing in Cardiology, Krakow, Poland, 9–12 September 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 393–396.
20. Antonucci, A.; De Rosa, R.; Giusti, A.; Cuzzolin, F. Robust classification of multivariate time series by imprecise hidden Markov models. *Int. J. Approx. Reason.* **2015**, *56*, 249–263. [CrossRef]
21. Pei, W.; Dibeklioğlu, H.; Tax, D.M.; van der Maaten, L. Multivariate time-series classification using the hidden-unit ogistic model. *IEEE Trans. Neural Networks Learn. Syst.* **2017**, *29*, 920–931. [CrossRef] [PubMed]
22. Ghassempour, S.; Girosi, F.; Maeder, A. Clustering multivariate time series using hidden Markov models. *Int. J. Environ. Res. Public Health* **2014**, *11*, 2741–2763. [CrossRef]
23. Dörpinghaus, J.; Schaaf, S.; Jacobs, M. Soft document clustering using a novel graph covering approach. *BioData Min.* **2018**, *11*, 11. [CrossRef]

24. Li, J.; Pedrycz, W.; Jamal, I. Multivariate time series anomaly detection: A framework of Hidden Markov Models. *Appl. Soft Comput.* **2017**, *60*, 229–240. [CrossRef]
25. Li, J.; Pedrycz, W.; Wang, X.; Liu, P. A Hidden Markov Model-based fuzzy modeling of multivariate time series. *Soft Comput.* **2023**, *27*, 837–854. [CrossRef]
26. Petropoulos, A.; Chatzis, S.P.; Xanthopoulos, S. A hidden Markov model with dependence jumps for predictive modeling of multidimensional time-series. *Inf. Sci.* **2017**, *412*, 50–66. [CrossRef]
27. Dörpinghaus, J.; Jacobs, M. Semantic Knowledge Graph Embeddings for biomedical Research: Data Integration using Linked Open Data. In Proceedings of the SEMANTiCS (Posters & Demos), Karlsruhe, Germany, 9–12 September 2019.
28. Dörpinghaus, J.; Stefan, A. Knowledge extraction and applications utilizing context data in knowledge graphs. In Proceedings of the 2019 Federated Conference on Computer Science and Information Systems (FedCSIS), Leipzig, Germany, 1–4 September 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 265–272.
29. Dörpinghaus, J.; Stefan, A.; Schultz, B.; Jacobs, M. Context mining and graph queries on giant biomedical knowledge graphs. *Knowl. Inf. Syst.* **2022**, *64*, 1239–1262. [CrossRef]
30. Dörpinghaus, J.; Klein, J.; Darms, J.; Madan, S.; Jacobs, M. SCAIView-A Semantic Search Engine for Biomedical Research Utilizing a Microservice Architecture. In Proceedings of the SEMANTiCS (Posters & Demos), Vienna, Austria, 10–13 September 2018.
31. Dörpinghaus, J.; Hübenthal, T.; Faber, J. A novelink prediction approach on clinical knowledge graphs utilising graph structures. In Proceedings of the 2022 17th Conference on Computer Science and Intelligence Systems (FedCSIS), Sofia, Bulgaria, 4–7 September 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 43–52.
32. Rabiner, L.R. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* **1989**, *77*, 257–286. [CrossRef]
33. Viterbi, A. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. Inf. Theory* **1967**, *13*, 260–269. [CrossRef]
34. Dempster, A.P.; Laird, N.M.; Rubin, D.B. Maximumikelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc. Ser. B (Methodol.)* **1977**, *39*, 1–22. [CrossRef]
35. Grandini, M.; Bagli, E.; Visani, G. Metrics for multi-class classification: An overview. *arXiv* **2020**, arXiv: 2008.05756.
36. Knuth, D.E. Backus normal form vs. backus naur form. *Commun. ACM* **1964**, *7*, 735–736. [CrossRef]
37. Virtanen, P.; Gommers, R.; Oliphant, T.E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nat. Methods* **2020**, *17*, 261–272. [CrossRef] [PubMed]
38. Klockgether, T.; Ludtke, R.; Kramer, B.; Abele, M.; Burk, K.; Schols, L.; Riess, O.; Laccone, F.; Boesch, S.; Lopes-Cendes, I.; et al. The natural history of degenerative ataxia: A retrospective study in 466 patients. *Brain* **1998**, *121 Pt 4*, 589–600. [CrossRef]
39. Schmitz-Hübsch, T.; du Montcel, S.T.; Baliko, L.; Berciano, J.; Boesch, S.; Depondt, C.; Giunti, P.; Globas, C.; Infante, J.; Kang, J.S.; et al. Scale for the assessment and rating of ataxia. *Neurology* **2006**, *66*, 1717–1720. [CrossRef]
40. Jacobi, H.; Rakowicz, M.; Rola, R.; Fancellu, R.; Mariotti, C.; Charles, P.; Durr, A.; Kuper, M.; Timmann, D.; Linnemann, C.; et al. Inventory of Non-Ataxia Signs (INAS): Validation of a new clinical assessment instrument. *Cerebellum* **2013**, *12*, 418–428. [CrossRef]
41. Reetz, K.; Dogan, I.; Hilgers, R.D.; Giunti, P.; Mariotti, C.; Durr, A.; Boesch, S.; Klopstock, T.; de Rivera, F.J.R.; Schols, L.; et al. Progression characteristics of the European Friedreich's Ataxia Consortium for Translational Studies (EFACTS): A 2 year cohort study. *Lancet Neurol.* **2016**, *15*, 1346–1354. [CrossRef]