Gaussian HMMs and Conditional GANs for Multi-Step Time Series Prediction

Minwoong Yoon

November 2025

Abstract

Building a model to predict time series data proves a challenge because of both the data's properties and model design complexity. Many existing approaches—such as RNNs, LSTMs, and autoregressive (AR) models—address sequential data, yet they are fundamentally deterministic and lack the generative flexibility of GANs [1,2]. As a result, designing a truly predictive and generative model remains an active area of research. In this work, we propose an HMM-GAN framework to explore these challenges directly. We first use a Hidden Markov Model (HMM), an unsupervised method, to preprocess the input. Which in turn, are fed into the GAN component, enabling generative modeling on a compact, temporally informed representation.

1 Introduction

It is known that EEG channels/electrodes are not independent where some electrodes capture the same brain wave frequency at certain positions [3]. However, we capture this channel to channel relationship with the Gaussian distribution assuming conditional independence given a hidden state which corresponds to a diagonal covariance matrix. This assumption, therefore, proposes the dependencies between channels thorugh the hidden states expressing the cross channel relationships.

1.1 Data

The dataset has already been preprocessed with a high-pass filter at 30Hz and a filter at 50 Hz. Also, each trial has already been cut into 60 seconds. Therefore, extreme outliers have already been omitted leaving only beta/gamma activity. We utilized three datasets, each with 31000 points, saved as train, validation, test data. Each of these data points were normalized and the dimensionality was reduced by using a tumbling window so that the mean of 100 data points can be expressed as one point or window.

2 Model

We utilize the 1st-order Markov Model to capture the temporal dependencies in time series data and assume the posterior probability of the dataset as a gaussian distribution s.t. the noisy, skewed, and heavy tailed characteristics of EEG data can be addressed. I.e even when the true distribution is not perfectly Gaussian, the HMM model proves a good local estimator to classify the stochastic nature of the data points. The emissions and the posterior states from the Gaussian HMM are then fed into the GAN's generator to create a new sequence of data points which are eventually distinguished by the GAN's discriminator to create a "highly" probabilitistic set of predictive outputs [1].

2.1 Problem Formulation

Our dataset $X_{1:t} = [x_1, x_2, x_3, ...x_t]$ is a list of vectors $x_t \in \mathbb{R}^{19}$(19 electrodes/ channels) with time step, t. By chain rule and Markov assumption, we propose the model's distribution:

$$x_t|(z_t = k) \sim \mathcal{N}(\mu_k, \sum\nolimits_k)$$

where z is the hidden state of the HMM. The Baulm-Welch algorithm iteratively computes the Expectation-Maximization (EM) to converge to the local optimum. The forward process is a recursive formula which addresses the temporal structure by $\alpha_t(i) = p(z_t = i, x_{1:t}|\theta)$ and the recursive backward process is $\beta_t(i) = p(x_{t+1:T}|z_t = i, \theta)$. The state posterior $\gamma_t(i) = p(z_t = k|x_{1:T}, \theta)$ calculates the probability of a sequence in state k given the observation and parameter values. The transition posterior $\xi_t(i,j) = P(z_t = i, z_{t+1} = j|x_{1:T}, \theta)$ captures the transition probabilities given x_t, θ. The state and transition posterior probabilities optimize the parameters((θ): prior $p(z_1 = k) = \gamma_1(k) = \pi_k$, transition A_{ij}, mean μ_k, and covariance $\sum_k$)in each iteration until the log-likelihood of the dataset ($lnP(x_t|\theta)$) converges[5].

Assuming the model's 3 hidden states $k \in \{1, 2, 3\}$ in $(z_t = k)$ and given our 19-D vectors(x_t) and proposed model distribution $\mathcal{N}(\mu_k, \text{Cov}_k^2)$, we compute $\alpha_t(i)$ and $\beta_t(i)$ to get our state posterior $\gamma_t(k)$ - mapping 19-D to 3-D. This 3-D vector(γ_t) provides the probability of our data point at a certain "brain state (k)." Therefore, at sequence length (L), our GAN prediction model has 3 inputs flattened to a one input vector:

$$latent(h) = \gamma_{t-L:t-1}$$

$$noise(z)$$

$$sequence = (x_{t-L:t-1})$$

which thereby becomes:

$$Input = [z, x_{t-L:t-1}, \gamma_{t-L:t-1}]$$

Given this input vector, we define a conditional generator G_ϕ and discriminator D_ψ. The generator maps the flattened input to a 19-D prediction of the next EEG window:

$$\hat{x}_t = G_\phi\big(z,\ x_{t-L:t-1},\ \gamma_{t-L:t-1}\big) \in \mathbb{R}^{19}.$$

The discriminator receives the same context together with either the real future emission x_t or the generated future $\hat{x}_t$, and outputs a scalar logit measuring how "real" the pair looks:

$$D_\psi\big(x_{t-L:t-1}, \gamma_{t-L:t-1}, x_t\big), \quad D_\psi\big(x_{t-L:t-1}, \gamma_{t-L:t-1}, \hat{x}_t\big) \in \mathbb{R}.$$

During training we alternate between updating the discriminator and the generator. Using a binary cross-entropy loss with logits, the discriminator is trained to distinguish real from generated futures:

$$\mathcal{L}_D(\psi) = \mathbb{E}\big[\,\ell_{\mathrm{BCE}}(D_\psi(x_{t-L:t-1}, \gamma_{t-L:t-1}, x_t), 1)\,\big] + \mathbb{E}\big[\,\ell_{\mathrm{BCE}}(D_\psi(x_{t-L:t-1}, \gamma_{t-L:t-1}, \hat{x}_t), 0)\,\big],$$

where ℓ_{BCE} is the binary cross-entropy loss and the expectations are taken over training windows and noise samples z.

The generator is trained with a combination of an adversarial term (trying to fool the discriminator) and a regression term (predicting the true next emission). Its objective is

$$\mathcal{L}_G(\phi) = \mathbb{E}\big[\,\ell_{\mathrm{BCE}}(D_\psi(x_{t-L:t-1}, \gamma_{t-L:t-1}, \hat{x}_t), 1)\,\big] + \lambda_{\mathrm{MSE}}\,\mathbb{E}\big[\,\|\hat{x}_t - x_t\|_2^2\big],$$

where $\lambda_{\mathrm{MSE}} > 0$ controls the trade-off between adversarial realism and one-step prediction accuracy.

At test time, given a new context $(x_{t-L:t-1}, \gamma_{t-L:t-1})$, we sample noise $z \sim \mathcal{N}(0, I)$ and obtain our final prediction for the next EEG window via

$$\hat{x}_t = G_\phi\big(z,\ x_{t-L:t-1},\ \gamma_{t-L:t-1}\big).$$

For multi-step forecasting, we iteratively append $\hat{x}_t$ to the context and roll the window forward, generating $\hat{x}_{t+1}, \hat{x}_{t+2}, \ldots$ in an autoregressive fashion [1,2,5].

2.2 Architecture

As seen in Figure 1, the model is composed of two blocks: G-HMM and a conditional GAN. The generator consists of two hidden layers(dimension=256) in a fully connected network. The discriminator is also an MLP but has 3 hidden layers of dimensions of size 256 and LeakyReLU activations. The input to the generator are the noise - Z, sequence $(x_{t-L:t-1})$, and state posterior $(\gamma_{t-L:t-1})$. We reuse the x_t and γ_t values to create the "real" value inputs to the discriminator. Then the discriminator outputs a logit value that represents the predicted probability the the generated/false sample is real.

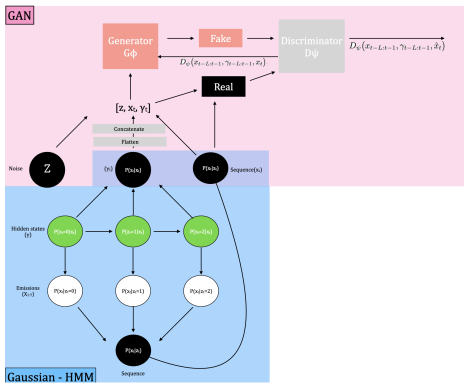

Figure 1: HMM-GAN Model

3 Results

3.1 Train

Training continues for 100 epochs in sliding windows of length $L = 30$ and batch size 32. During training, the discriminator loss $\mathcal{L}_D$ quickly converges to a range of $\approx$ 1.3 1.4, while the adversarial component of the generator loss stabilizes around $\approx$ 0.68 0.70. This is consistent with a near-equilibrium state, where the discriminator cannot reliably distinguish real from generated futures, and its accuracy is close to random guessing.

The mean-squared-error term $\mathbb{E}\|\hat{x}_t - x_t\|_2^2$ decreases steadily over epochs and reaches a small value (on the order of 10^{-2} in normalized units). This indicates that the generator behaves as a strong one-step regressor, producing predictions that are close to the next observed EEG window in an average squared-error sense.

$\hat{x}_t = x_{t-1}$ for all t obtains a validation MSE of about 0.30, so the proposed HMM–GAN reduces the error by roughly 26% relative to this naive predictor. This gap indicates that the model is learning non-trivial local dynamics beyond merely copying the last observed window, even though the adversarial term remains close to the regime where the discriminator cannot reliably separate real from generated futures.

3.2 Test

3.2.1 One-Step-Test

To evaluate one-step performance, the dataset is partitioned into sliding windows and the model appends each fake output into a set. Then, for each context window, the generator produces a single-step prediction $\hat{x}_t$, which is compared to the true future x_t.

The loss was calculated by the mean-squared error (MSE) averaged across

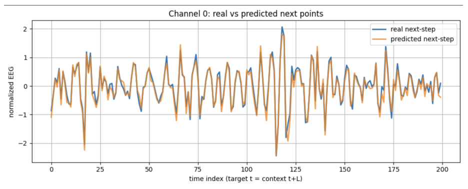

Figure 2: One-step Forecast

time, channels, and test windows. Qualitatively, plotting individual channels shows that the predicted next window follows the local trend of the true EEG, especially for slowly varying components. Peaks and rapid fluctuations are sometimes underestimated, reflecting the bias of MSE toward smoother predictions and regression toward the mean.

The binary cross-entropy loss of the discriminator stabilized around 1.3–1.4 and the adversarial loss of the generator around 0.69, indicating that the discriminator could not reliably distinguish real from generated futures and operated close to random guessing. This is because of the fact that the model is heavily supervised by the one-step MSE term where the model learns a near-deterministic regression from context to the next EEG window. As a result, the distributions of real and generated one-step futures become very similar, confusing the discriminator. Tuning the MSE weight and learning rates did not change this outcome, suggesting that the limitation is a structural (single-step, mean-squared objective on a small dataset) rather than a parametric problem.

3.2.2 Multi-Step-Test

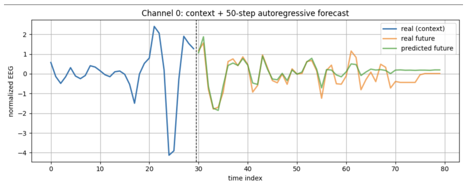

Figure 3: Multi-step Forecast

For multi-step evaluation, the model is run in an autoregressive fashion. Given a context window of length $L = 30$, we first use the trained generator to predict $\hat{x}_t$ from $(x_{t-L:t-1}, \gamma_{t-L:t-1})$. This prediction is then appended to the context and the oldest time step is discarded, forming a new window $(\hat{x}_{t-L+1:t}, \gamma_{t-L+1:t-1})$ from which we generate $\hat{x}_{t+1}$. Repeating this procedure yields a forecast trajectory $\hat{x}_{t:t+H-1}$ over a horizon H (e.g., $H = 50$ or $H = 200$ windows).

Quantitatively, we compute the multi-step MSE between the generated trajectory and the true future $x_{t:t+H-1}$, averaged over channels and starting points in the validation sequence. Qualitatively, as shown in Figure 3, the model is able to follow the short-term trend of the EEG signal for the first few steps, but the predictions gradually become smoother and lose high-frequency oscillatory structure as the time increases. This behavior is consistent with the training: because the model is only penalized for one-step errors, it learns a locally accurate but increasingly low forecast when implemented autoregressively. In other words, the HMM–GAN behaves like a strong one-step regressor, leading to flattening of the predicted values over many steps.

3.3 Evaluation

The HMM-GAN achieves low one-step MSE on the normalized EEG data and produces plausible single-step forecasts. The adversarial component encourages the generator to stay on the manifold of realistic EEG windows, while the HMM posteriors provide a compressed summary of latent "brain states" that regularizes the input and reduces dimensionality [3, 4].

At the same time, the current architecture behaves largely like a deterministic conditional regressor: for a fixed context, different noise samples z produce very similar predictions. This indicates that, under the chosen loss $\mathcal{L}G$ and λMSE, the model tends to ignore the noise in order to minimize MSE, which shows less generative ability. Furthermore, because the model is only trained with a one-step loss, long time forecasts flatten and lose the oscillatory structure of real EEG.

These observations suggest that the proposed approach is a useful first step toward generative forecasting, but that other modeling choices are required to fully capture the stochastic nature of the latent space and to model the dynamics over longer periods of time.

4 Extension

All work from research to implementation was done on my own. There are much better architectures such as the Time-Gan, but in this attempt we explore a simple approach to better comprehend how we can utilize supervised and unsupervised techniques to leverage complex data. In the future, I will implement a RNN generator s.t. the dynamics of the data are learned.

5 References

[1] Yoon, Jinsung, Daniel Jarrett, and Mihaela Van der Schaar. "Time-series generative adversarial networks." Advances in neural information processing systems 32 (2019).

[2] Goodfellow, Ian J., et al. "Generative adversarial nets." Advances in neural information processing systems 27 (2014).

[3] Hartmann, Kay Gregor, Robin Tibor Schirrmeister, and Tonio Ball. "EEG-GAN: Generative adversarial networks for electroencephalograhic (EEG) brain signals." arXiv preprint arXiv:1806.01875 (2018).

[4] Habashi, Ahmed G et al. "Generative adversarial networks in EEG analysis: an overview." Journal of neuroengineering and rehabilitation vol. 20 (2023)

[5] Wang, Zhaoran, et al. "High dimensional expectation-maximization algorithm: Statistical optimization and asymptotic normality." arXiv preprint arXiv:1412.8729 (2014).