Symbol-Level Online Channel Tracking for Deep Receivers

Ron Aharon Finish, Yoav Cohen, Tomer Raviv, and Nir Shlezinger

***Abstract*—Deep neural networks (DNNs) allow digital receivers to operate in complex environments by learning from data corresponding to the channel input-output relationship. Since communication channels change over time, DNN-aided receivers may be required to retrain periodically, which conventionally involves excessive pilot signaling at the cost of reduced spectral efficiency. In this paper, we study how one can obtain data for retraining deep receivers without sending pilots or relying on specific protocol redundancies, by combining self-supervision with active learning concepts. We focus on the recently proposed ViterbiNet receiver, which integrates into the Viterbi algorithm a DNN for learning the channel. To enable self-supervision, we use the soft-output Viterbi algorithm to evaluate the decision confidence for each of the detected symbols in a given word. Then, to overcome learning with erroneous data, we choose a subset of the recovered symbols to be used for retraining via active learning. The proposed method selects decision-directed data whose confidence is not too low to result in inaccurate labeling, yet not too high to preserve sufficient diversity of the data. We demonstrate that self-supervised symbol-level training yields a performance within a small gap of the Viterbi algorithm with instantaneous channel knowledge.**

***Index terms*— Active learning, self-supervision, Viterbi algorithm.**

I. Introduction

Recent years have witnessed a growing research attention in the application of deep learning in digital communications systems and receiver design [1]–[5]. Unlike conventional receiver algorithms, which rely on principled mathematical models of the signal transmission, propagation, and reception, deep neural networks (DNNs) are model-agnostic, and learn their mappings from data. DNN-aided receivers can operate efficiently in scenarios where the channel model is unknown, highly complex, or difficult to optimize for [6].

Despite their potential, deep learning solutions are subject to several challenges that limit their applicability in important communication scenarios. A fundamental difference between digital communications and traditional deep learning applications, such as computer vision, stems from the dynamic nature of communication systems [5]. DNNs consist of highly-parametrized models that can represent a broad range of mappings. As such, massive data sets are typically required to learn a desirable mapping. The dynamic nature of communication channels implies that the statistical model can change considerably over time, and thus a DNN trained for a given channel may no longer perform well on a future channel. DNN-aided receivers are thus likely to require frequent retraining, at the cost of degraded spectral efficiency due to pilot transmissions.

Various strategies have been proposed to facilitate the application of DNNs-aided receivers in dynamic channels. The first type avoids retraining, attempting instead to learn a single mapping that is applicable to a broad range of channel conditions. This class of methods includes the straightforward approach, commonly referred to as *joint learning*, of training a DNN using data corresponding to a broad set of expected channel conditions [4], [7]. An additional related method is to train in advance a different network for each expected statistical model, and combine them as a deep ensemble [8]. However, these strategies typically require large training sets, and deviating from the training setup can degrade performance [9].

The alternative strategy is to track the channel variations. This can be achieved by providing the DNN with a conventional model-based channel estimate [10]–[13]. However, such channel estimation involves imposing a relatively simple model on the channel, e.g., a linear Gaussian model, which may be inaccurate in some setups and does not exploit the model-agnostic nature of DNNs. When using model-agnostic DNN-aided receivers, tracking variations involves periodically retraining the network. The volumes of data one can obtain in real-time are limited and are not at the scale of data sets typically used for training DNNs. To provide data for retraining in the detection scenario, one must either transmit frequent pilots, as is usually done in communication, or use decoded data for training. The former is simpler but significantly reduces the data transmission. The latter can be achieved by training with successfully decoded forward error correction (FEC) codewords, as in [14], [15]. This implies however that retraining is possible only when the codewords are successfully decoded, thus not allowing to track channels in low signal-to-noise ratio (SNR) conditions as observed in [14], [16]. Alternatively, one can provide a measure of confidence for each symbol, and select the ones with the highest confidence for retraining, as suggested in [17], [18]. Such symbol-level online training does not rely on pilots, FEC decoding, or any form of protocol specifications. Nonetheless, there is no guarantee that the decoded symbols indeed provide accurate labeling. Furthermore, as active learning theory suggests [19], high-confidence data are often of limited contribution for training DNNs.

In this work we propose a self-supervised symbol-level active learning scheme for DNN-aided receivers operating in blockwise time-varying channels. We focus on finite-memory channels and use the ViterbiNet DNN-aided receiver [14] to implement the soft-output Viterbi algorithm (SOVA). To allow channel tracking in a model-agnostic fashion, we adopt the active learning approach to enable self-supervised online training, using the SOVA outputs as the measure of confidence. We innovate a selective-sampling scheme by which we only select a proportion of the detected symbols in each block for retraining. The selection is focused on intermediate confidence symbols: Low-valued symbols are mostly noisy, while high-valued ones are error-free thus do not contribute to the learning. The resulting mechanism is shown to facilitate online learning by training with only a small number of symbols in a rate-loss free manner, while reducing the symbol error rate (SER) compared to previous methods.

The rest of this paper is organized as follows: Section II details the system model. Section III presents our symbol-level online training approach. Experimental results are presented in Section IV.

II. System Model

A. Channel Model

We consider communications over causal finite-memory blockwise-stationary channels. The channel output depends on the last $L > 0$ transmitted symbols, where L is the memory length. The channel is constant within a block of B channel uses, representing the coherence duration. Let $S_{i,j} \in \mathcal{S}$, with $|\mathcal{S}| = M$, be the symbol transmitted from constellation $\mathcal{S}$ at the ith time instance $i \in \{1, 2, \ldots, B\} := \mathcal{B}$ of the jth block. The corresponding channel output, denoted $\boldsymbol{Y}_{i,j}$, is given by a stochastic function of the last L transmitted symbols $\bar{\boldsymbol{S}}_{i,j} := [S_{i-L+1,j}, \ldots, S_{i,j}]^T$. Specifically, by defining the jth transmitted block as $\boldsymbol{S}_j^B := \{S_{i,j}\}_{i\in\mathcal{B}}$ and its corresponding observations as $\boldsymbol{Y}_j^B := \{\boldsymbol{Y}_{i,j}\}_{i\in\mathcal{B}}$, the conditional distribution of the channel output given its input satisfies

$$p_{\boldsymbol{Y}_j^B|\boldsymbol{S}_j^B}\left(\boldsymbol{y}_j^B|\boldsymbol{s}_j^B\right) = \prod_{i=1}^{B} p_{\boldsymbol{Y}_{i,j}|\bar{\boldsymbol{S}}_{i,j}}\left(\boldsymbol{y}_{i,j}|\bar{\boldsymbol{s}}_{i,j}\right). \tag{1}$$

The authors are with the School of Electrical and Computer Engineering, Ben-Gurion University of the Negev, Beer-Sheva, Israel (e-mail: {finishr, yoco}@post.bgu.ac.il, tomerraviv95@gmail.com, nirshl@bgu.ac.il).

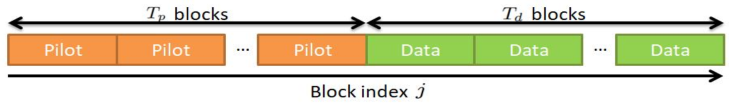

Fig. 1. Transmission model. The channel is constant within each block and changes across blocks.

In (1), $\boldsymbol{y}_{i,j}$ and $\bar{\boldsymbol{s}}_{i,j}$ are the realizations of the random variables (RVs) $\boldsymbol{Y}_{i,j}$ and $\bar{\boldsymbol{S}}_{i,j}$, respectively, while $S_{i,j} \equiv 0$ for $i < 0$. Each symbol $S_{i,j}$ is uniformly distributed over the set $\mathcal{S}$ of M constellation points.

B. Problem Formulation

We consider the sequential transmission of $T = T_p + T_d$ blocks as illustrated in Fig. 1. The first T_p blocks are known pilots that are used for the initial training of a DNN-aided detector. The next T_d blocks contain data, where T_d can be much larger that T_p. Each block is comprised of B uncoded symbols.

Our goal is to design a mechanism which allows DNN-aided receivers to track the temporal variations in the underlying channel under the transmission protocol of Fig. 1. We utilize the ViterbiNet symbol detector, recalled in the following section, being a relatively low-parametrized DNN-based receiver suitable for channels of the form (1). Nonetheless, the scheme proposed is not unique to ViterbiNet, and can be combined with other DNN-aided receivers.

C. ViterbiNet

The ViterbiNet equalizer, proposed in [14], is a data-driven implementation of the Viterbi detector [20] for finite-memory channels of the form (1) . ViterbiNet does not require prior knowledge of the channel conditional distributions $p_{\boldsymbol{Y}_j^B|\boldsymbol{S}_j^B}$.

The classic Viterbi equalizer, which ViterbiNet learns to implement, solves the maximum likelihood sequence detection problem for each data block j

$$\hat{\boldsymbol{s}}_j^B\left(\boldsymbol{y}_j^B\right) = \mathop{\arg\min}_{\boldsymbol{s}^B \in \mathcal{S}^B} \sum_{i=1}^{B} c_{i,j}(\bar{\boldsymbol{s}}), \tag{2}$$

where the *path cost* $c_{i,j}(\bar{\boldsymbol{s}})$ is defined as

$$c_{i,j}(\bar{\boldsymbol{s}}) := -\log p_{\boldsymbol{Y}_{i,j}|\bar{\boldsymbol{S}}_{i,j}}\left(\boldsymbol{y}_{i,j}|\bar{\boldsymbol{s}}_i\right). \tag{3}$$

The Viterbi equalizer solves (2) via dynamic programming, iteratively updating (3) for each state $\bar{\boldsymbol{s}} \in \mathcal{S}^L$ and for each time instance $i = 1, 2, \ldots, B$. ViterbiNet implements Viterbi detection in a data-driven fashion by training a DNN to provide a parametric estimate of the likelihood function $p_{\boldsymbol{Y}_{i,j}|\bar{\boldsymbol{S}}_{i,j}}(\boldsymbol{y}|\bar{\boldsymbol{s}})$, which is denoted as $\hat{P}_{\boldsymbol{\varphi}}(\boldsymbol{y}|\bar{\boldsymbol{s}})$, where $\boldsymbol{\varphi}$ are the model parameters. See [14] for more details.

III. Symbol-Level Online Training

DNN-aided receivers operate most reliably when tested under the same statistical channel conditions observed during training. In particular, ViterbiNet requires its model parameters $\boldsymbol{\varphi}$ to be set such that $\hat{P}_{\boldsymbol{\varphi}}$ is an accurate estimate of $p_{\boldsymbol{Y}_{i,j}|\bar{\boldsymbol{S}}_{i,j}}$ (possibly up to some constant factor [21]). The fact that the channel conditions change with the block index j indicates that the parameter vector $\boldsymbol{\varphi}$ should track the time-varying channel. To enable channel tracking in the absence of periodic pilots without relying on successful FEC decoding, we propose a symbol-level self-supervision algorithm based on active learning principles. We begin by discussing its rationale in Subsection III-A and formulating the measure of confidence in Subsection III-B. Then, we identify the key properties of symbol-level data selection and present the active learning algorithm in Subsection III-C. Finally, we discuss its properties in Subsection III-D

A. Active Learning Rationale

Our proposed symbol-level self-supervision approach utilizes the decisions made by the DNN-aided receiver $\hat{\boldsymbol{s}}_j^B$, along with its corresponding channel outputs $\boldsymbol{y}_j^B$ for adapting the DNN-aided receiver at block index j. This approach allows to track the channel variations assuming that they are relatively smooth, such that a DNN-aided receiver trained using data from the jth block can reliably detect the following block. The receiver has no knowledge which of the detection symbols in $\hat{\boldsymbol{s}}_j^B$ corresponds to the true transmitted symbol, and detection errors are expected to have a harmful effect on the retrained model. This motivates the need for a mechanism to sample a subset of the detected symbols for online training.

Active learning is a supervised learning method, which deals with an oracle that actively chooses the samples from a large pool of data to feed a model [22]. The oracle can either be a human annotator, or a machine based one. The conventional motivation for feeding a model with a selected subset of the data is that a properly queried batch can benefit the training more than using the entire unfiltered batch. A key ingredient in implementing active learning is the introduction of a metric that reflects the informativeness of each data sample. Once such a metric is determined, active learning operates in an iterative fashion, where at each step, a batch is filtered based on the metric value that each of its elements has, and then queried for training the model [19], [22]. Considering the example from [18], the authors suggest three measures: The hamming distance, a discrete measure; A soft reliability that measures the deviation of the channel output probabilities from the corresponding transmitted bits; And finally, the mean bit cross entropy.

B. Confidence Measure

In order to apply active learning based data selection, one has to provide some measure for the usefulness of the detected data. Since in our case the data may include detection errors, we formulate this measure based on the output of ViterbiNet used for implementing SOVA, as such outputs can be related to the detection confidence.

To formulate the SOVA output, we omit the block index j, and define the minimal cost state for the ith time instance as $\bar{\boldsymbol{s}}_i^* \triangleq \arg\min_{\bar{\boldsymbol{s}} \in \mathcal{S}^L} c_i(\bar{\boldsymbol{s}})$, and the second-to-minimal cost state as $\bar{\boldsymbol{s}}_i^{**} \triangleq \arg\min_{\bar{\boldsymbol{s}} \in \mathcal{S}^L / \bar{\boldsymbol{s}}_i^*} c_i(\bar{\boldsymbol{s}})$. Next, we define the gap

$$g_i \triangleq |c_i(\bar{\boldsymbol{s}}_i^*) - c_i(\bar{\boldsymbol{s}}_i^{**})|, \tag{4}$$

and argue that it reflects the confidence in the decision, as it typically does in the model-based SOVA [23, Ch. 9.4].

To empirically validate that higher values of g_i indeed correspond to more confident detection, we evaluate the error rate of ViterbiNet while considering outputs from different percentiles of the corresponding values of g_i in each detected block. In the experiment, whose results are reported in Fig. 2, we consider the numerical setup described in Section IV, and observe the confidence measure and the detection success rate over the first 25 data blocks for different SNR values. Observing Fig. 2, we note that higher soft output values produced by ViterbiNet indeed reflect on more confident decisions, and as the gap increases, so is the confidence in the selected symbol. For instance, for SNR of 2 dB, we observe that while the success rate of all decisions is less than 0.7, for the decisions with the 10% highest values of g_i, the success rate surpasses 0.9. This indicates that g_i can be used as a measure of confidence by ViterbiNet.

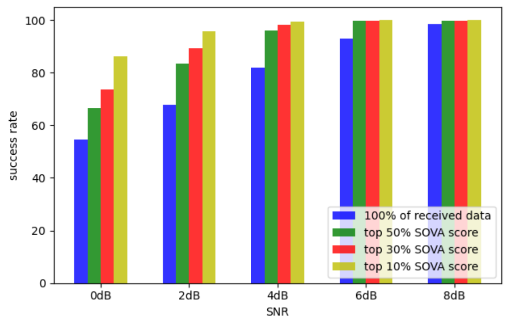

Fig. 2. Success rate of ViterbiNet by SNR for different percentiles of the SOVA score g_i (4).

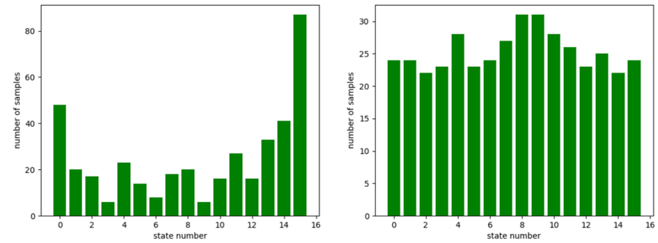

Fig. 3. Symbol diversity among the $2^4 = 16$ states for decisions in the highest soft-outputs percentiles of $80\% - 100\%$ (left) and $30\% - 50\%$ (right).

C. Self-Supervised Active Learning

Since the SOVA score (4) reflects on confidence, we use this value to select decisions to be used for online training. Given a channel output block $\boldsymbol{y}_j^B$, ViterbiNet with weights $\boldsymbol{\varphi}_j$ maps it into the detected symbols $\hat{\boldsymbol{s}}_j^B$ and confidence measures $\boldsymbol{g}^B \triangleq [g_1, \ldots, g_B]^T$. Then, $\boldsymbol{g}^B$ is used to select a subset of $\{(\boldsymbol{y}_{i,j}, \hat{s}_{i,j})\}_{i=1}^B$ for training $\boldsymbol{\varphi}_j$ into $\boldsymbol{\varphi}_{j+1}$. This selection accounts for the following considerations:

1) Correctness of Decisions: When using symbol-level decisions for retraining, one is inherently sensitive to incorrect detection. Since g_i is related to the correctness in detecting $s_{i,j}$, we wish to avoid using decisions with low confidence values for self-supervision. Consequently, we do not use decisions in the lower percentiles of the SOVA in each block for retraining. We fix some lower percentile $\ell \in (0, 1]$ and its corresponding threshold P_ℓ, such that all sample pairs $(\boldsymbol{y}_{i,j}, \hat{s}_{i,j})$ for which $g_i < P_\ell$ are not used for retraining.

While confidence is important when training in a self-supervised manner, in active learning one has to account for another important trait: the diversity of the samples [24]. Here, we consider two forms of data diversity: reliability diversity and symbol diversity.

2) Reliability Diversity: Data samples which are identified as most reliable in terms of confidence are not necessarily the most informative for training. Specifically, samples with high confidence values often correspond to channel outputs with lesser magnitude noise realizations, compared with low confidence samples. As suggested by active learning theory [22] and observed in previous applications of active learning for communication (which did not use self-supervision) [18], samples with more dominant noise realization are more informative for training compared to less noisy ones.

3) Symbol Diversity: To properly train the DNN, the training set must be balanced in terms of the available labels. Such diversity is crucial to avoid unbalanced decisions, and the selected data should cover the entire distribution of states, avoiding over-training on a specific state. However, when communicating over finite-memory channels, some states (i.e., combinations of symbols) may be easier to detect compared to others. Thus, selecting data samples with high confidence values may result in an unbalanced data set.

To empirically assert this statement, we depict in Fig. 3 the distributions of the symbols detected by ViterbiNet for the numerical setup reported in Section IV for two different percentiles of the confidence values $\{g_i\}$: decisions with the 20% highest confidence scores, i.e., $P_{0.8} \leq g_i \leq P_1$, and decisions around the median selected as those satisfying $P_{0.3} \leq g_i \leq P_{0.5}$. We observe in Fig. 3 that the most reliable decisions result in unbalanced labels geared towards states '0' and '15' (both correspond to the transmitter sending the same symbol consecutively, being easier to detect for the simulated channel model). However, data labels from the median confidence percentile are approximately uniformly distributed over the possible states, hence the resulting data is relatively balanced.

Algorithm 1: Self-Supervised Active Learning

Initialization: empty buffer $\mathcal{Q}$
Input : current detector $\boldsymbol{\varphi}_j$
received channel-block $\boldsymbol{y}_j^B$
ℓ, u lower and upper percentiles
Output : improved model $\boldsymbol{\varphi}_{j+1}$

1 **Self-Supervised Active Learning** $(\boldsymbol{\varphi}_j, \boldsymbol{y}_j^B, \ell, u)$
2 $c_{i,j}(\bar{\boldsymbol{s}}) \leftarrow$ calculate by (3), $(\bar{\boldsymbol{s}}, i) \in \mathcal{S} \times \mathcal{B}$;
3 $\hat{\boldsymbol{s}}_j^B \leftarrow$ calculate by (2);
4 $\boldsymbol{g}^B \leftarrow$ calculate by (4);
5 $P_\ell, P_u \leftarrow$ thresholds of confidence ℓ, u percentiles;
6 **for** i *in* $\mathcal{B}$ **do**
7 **if** $P_\ell \leq g_i \leq P_u$ **then**
8 add $(\boldsymbol{y}_{i,j}, \hat{s}_{i,j})$ to Q;
9 **end**
10 $\boldsymbol{\varphi}_{j+1} \leftarrow$ train model $\boldsymbol{\varphi}_j$ using data $\mathcal{Q}$;
11 **return** $\boldsymbol{\varphi}_{j+1}$

---

4) Self-Supervised Active Learning Algorithm: Based on the aforementioned considerations, we propose to select the data based on by the confidence measure $\boldsymbol{g}^B$ by fixing both a lower percentile threshold (to boost correctness), and a high percentile threshold (to encourage diversity). The pseudo-code of our approach is summarized in Algorithm 1: For each received block $\boldsymbol{y}_j^B$, the current model $\boldsymbol{\varphi}_j$ is used to detect the symbols (line 3) and compute confidence (line 4). The batch used for retraining the model to track the channel, denoted $\mathcal{Q} \subset \{(\boldsymbol{y}_{i,j}, \hat{s}_{i,j})\}_{i=1}^B$ is queried to select intermediate confidence values, determined by the pre-defined percentiles ℓ, u (line 7). Finally, we update the model into $\boldsymbol{\varphi}_{j+1}$ by training using the selected data $\mathcal{Q}$ while using $\boldsymbol{\varphi}_j$ as warm start (line 10).

D. Discussion

Algorithm 1 combines decision-directed self-supervision with the active learning methodology, selecting a subset of the symbols for online training at each time-step. This operation differs from previously proposed approaches for self-supervised online training, e.g., [14], [16], [25], which relied on FEC codes, requiring the entire block to be correctly decoded to produce decision-directed data. Our symbol level active learning scheme identifies subsets of the decisions which are useful for self-supervision regardless of presence of channel coding, allowing it to be applied in uncoded transmissions and in regimes where FEC decoding fails, e.g., low SNRs, as we numerically demonstrate in Section IV.

Algorithm 1 involves the fixing of two hyperparameters, (ℓ, u), which dictate the percentiles of confidence scores by which data is selected. The setting of (ℓ, u) balances the considerations of correctness and diversity, as well as the size of the selected data set $\mathcal{Q}$. In our numerical study we observe that ViterbiNet can be reliably adapted when $|\mathcal{Q}|$ is of the order of merely a few hundreds of symbols, owing to its hybrid model-based/data-driven architecture [26]. Nonetheless, while we formulate Algorithm 1 for ViterbiNet, the

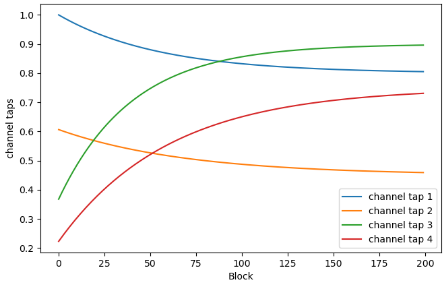

Fig. 4. Channel coefficients versus block index.

same mechanism can be used with alternative DNN-based receivers (at the possible modification of the confidence measure), involving larger numbers of trainable parameters. To obtain more data, one can either use a larger gap between u and ℓ, as well as consider retrain once every multiple blocks (assuming that the variations are sufficiently smooth). Alternatively, large models can be tuned to allow training with small datasets by incorporating meta-learning tools, as in [9], [16]. We leave these extensions for future investigation.

IV. Experimental Results

We simulate a scalar finite-memory linear Gaussian channel with $L = 4$ channel taps. At the jth block, the channel is given by

$$Y_{i,j} = \sum_{l=1}^{L} h_{l,j} S_{i-l+1,j} + W_{i,j}, \tag{5}$$

where $W_{i,j}$ is Gaussian noise with variance σ_w^2, and the $\{h_{l,j}\}$ are the channel taps, whose temporal variations in the block index j follow an exponential decaying/rising profile as depicted in Fig. 4. The symbols are generated from a BPSK constellation, i.e., $\mathcal{S} = \{\pm 1\}$, and each block is comprised of 2000 symbols, except for the first pilots block ($j = 0$), which consists of 5000 symbols. The overall transmission includes $T_p = 1$ pilot block and $T_d = 25$ data blocks[1].

We implement ViterbiNet using an internal DNN with two fully-connected layers: a 1×75 layer followed by 75×16 layer, using intermediate ReLU activation functions and softmax output layer. The network is trained using the Adam optimizer to minimize the cross-entropy loss. We first train using the pilot symbols. Then, we use the following approaches for tracking the time-varying channel:

- *No train* - no online training.
- *Median training* - Algorithm 1 with $(\ell, u) = (0.3, 0.5)$.
- *Low score training* - Algorithm 1 with $(\ell, u) = (0, 0.2)$.
- *High score training* - Algorithm 1 with $(\ell, u) = (0.8, 1)$.
- *Coded training* - the self-supervised online training from [14]. Each block is encoded with a Reed-Solomon [255,250] code. When the average bit error rate is less than 2%, it is trained on the given block.
- *Genie* - online training with the true data $\boldsymbol{y}_j^B, \boldsymbol{s}_j^B$.
- *Genie* 20% - online training with a random 20% of $\boldsymbol{y}_j^B, \boldsymbol{s}_j^B$.

The high score training scheme bears similarity to the online training approach adopted in [17]. Nonetheless, here we consider much shorter blocks of merely $B = 2000$ symbols, while [17] used blocks of over $2.8 \cdot 10^4$ symbols, namely, we focus on channels which vary more rapidly compared to those considered in [17].

We first evaluate the instantaneous average SER achieved for the channel (5) with SNR (defined as $1/\sigma_w^2$) of 6 dB. The performance is compared with the application of the Viterbi algorithm which knows the channel at each time instance. The results, depicted in Fig. 5, demonstrate that the proposed median training Algorithm 1, which operates on a decision-directed manner, achieves similar SER

[1]The source code is available online at https://github.com/yoavchoen/Symbol-Level-Online-Training.

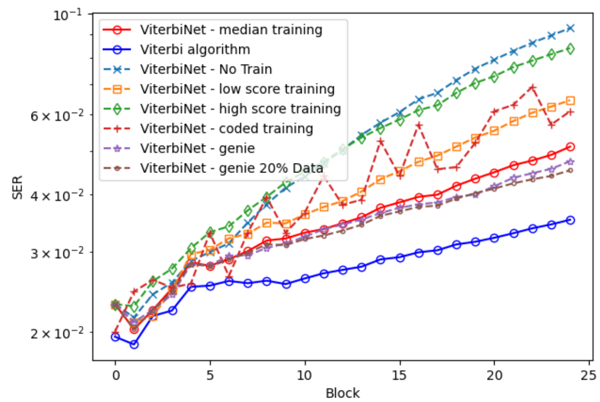

Fig. 5. Time-averaged SER versus block index for SNR of 6 dB.

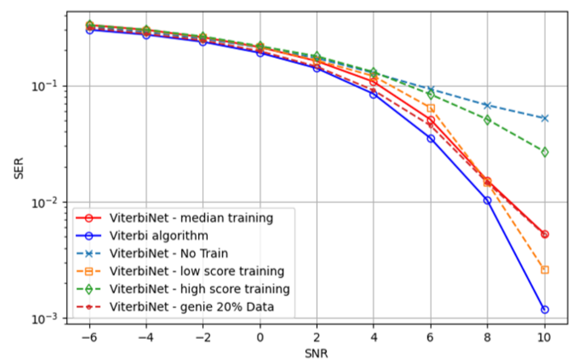

Fig. 6. SER after $T_d = 25$ blocks versus SNR.

values as those obtained when retraining VitebiNet with genie-provided true decisions for the entire block. It is also observed that the selection plays a crucial role in the success of self-supervised online training; taking the high confidence samples, as proposed in [17], results in an insufficiently diverse data for the considered short blocklength, achieving similar accuracy to not retraining at all. Finally, we notice that our method surpasses codeword-level self-supervision as occasional decoding failures occur at this SNR.

We next evaluate the SER averaged over all $T_d = 25$ data blocks, versus SNR. The results, depicted in Fig. 6, show that Algorithm 1 allows ViterbiNet to achieve SER values within a small gap of approximately 1 dB SNR compared to the model-based Viterbi algorithm operating with instantaneous channel state information. This is achieved with only an initial pilot transmission, without requiring knowledge of neither the channel model nor its parameters (except for the fact that is has finite memory), and while relying solely on self-supervision for channel tracking. It is also noted that in high SNRs, where most decisions are correct, using the diverse low score training is most beneficial. These observations strengthen the need to account for the multiple considerations detailed in Section III.

V. Conclusions

We proposed an active learning based mechanism for symbol-level online training of DNN-aided receivers. To that aim, we identified the key considerations in self-supervised training of deep receivers as decision correctness, symbol diversity, and reliability diversity. To implement data selection using the ViterbiNet model, we utilized the SOVA score as an indication the detection confidence of each symbol, and proposed an algorithm for selecting decisions for training. Our numerical results demonstrate the proposed algorithm achieves channel tracking in rapidly time-varying channels, approaching the performance of Viterbi detection with full channel knowledge.

REFERENCES

[1] D. Gündüz, P. de Kerret, N. D. Sidiropoulos, D. Gesbert, C. R. Murthy, and M. van der Schaar, "Machine learning in the air," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2184–2199, 2019.

[2] N. Farsad, N. Shlezinger, A. J. Goldsmith, and Y. C. Eldar, "Data-driven symbol detection via model-based machine learning," *arXiv preprint arXiv:2002.07806*, 2020.

[3] A. Balatsoukas-Stimming and C. Studer, "Deep unfolding for communications systems: A survey and some new directions," *arXiv preprint arXiv:1906.05774*, 2019.

[4] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. on Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, 2017.

[5] L. Dai, R. Jiao, F. Adachi, H. V. Poor, and L. Hanzo, "Deep learning for wireless communications: An emerging interdisciplinary paradigm," *IEEE Wireless Commun.*, vol. 27, no. 4, pp. 133–139, 2020.

[6] N. Farsad and A. Goldsmith, "Neural network detection of data sequences in communication systems," *IEEE Trans. Signal Process.*, vol. 66, no. 21, pp. 5663–5678, 2018.

[7] J. Xia, D. Deng, and D. Fan, "A note on implementation methodologies of deep learning-based signal detection for conventional MIMO transmitters," *IEEE Trans. Broadcast.*, vol. 66, no. 3, pp. 744–745, 2020.

[8] T. Raviv, N. Raviv, and Y. Be'ery, "Data-driven ensembles for deep and hard-decision hybrid decoding," *arXiv preprint arXiv:2001.06247*, 2020.

[9] S. Park, H. Jang, O. Simeone, and J. Kang, "Learning to demodulate from few pilots via offline and online meta-learning," *IEEE Trans. Signal Process.*, vol. 69, pp. 226–239, 2020.

[10] H. He, C.-K. Wen, S. Jin, and G. Y. Li, "Model-driven deep learning for MIMO detection," *IEEE Trans. Signal Process.*, vol. 68, pp. 1702–1715, 2020.

[11] M. Honkala, D. Korpi, and J. M. Huttunen, "DeepRx: Fully convolutional deep learning receiver," *IEEE Trans. Wireless Commun.*, 2021.

[12] M. Khani, M. Alizadeh, J. Hoydis, and P. Fleming, "Adaptive neural signal detection for massive MIMO," *IEEE Trans. Wireless Commun.*, vol. 19, no. 8, pp. 5635–5648, 2020.

[13] M. Goutay, F. A. Aoudia, J. Hoydis, and J.-M. Gorce, "Machine learning for MU-MIMO receive processing in OFDM systems," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 8, pp. 2318–2332, 2021.

[14] N. Shlezinger, N. Farsad, Y. C. Eldar, and A. J. Goldsmith, "ViterbiNet: A deep learning based Viterbi algorithm for symbol detection," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3319–3331, 2020.

[15] C.-F. Teng and Y.-L. Chen, "Syndrome enabled unsupervised learning for neural network based polar decoder and jointly optimized blind equalizer," *IEEE Trans. Emerg. Sel. Topics Circuits Syst.*, 2020.

[16] T. Raviv, S. Park, N. Shlezinger, O. Simeone, Y. C. Eldar, and J. Kang, "Meta-ViterbiNet: Online meta-learned Viterbi equalization for non-stationary channels," in *Proc. IEEE ICC*, 2021.

[17] L. Sun, Y. Wang, A. L. Swindlehurst, and X. Tang, "Generative-adversarial-network enabled signal detection for communication systems with unknown channel models," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 47–60, 2020.

[18] I. Be'ery, N. Raviv, T. Raviv, and Y. Be'ery, "Active deep decoding of linear codes," *IEEE Trans. Commun.*, vol. 68, no. 2, pp. 728–736, 2019.

[19] Y. Fu, X. Zhu, and B. Li, "A survey on instance selection for active learning," *Knowledge and information systems*, vol. 35, no. 2, pp. 249–283, 2013.

[20] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inf. Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[21] N. Shlezinger, N. Farsad, Y. C. Eldar, and A. J. Goldsmith, "Learned factor graphs for inference from stationary time sequences," *arXiv preprint arXiv:2006.03258*, 2020.

[22] B. Settles, "Active learning literature survey," 2009.

[23] J. M. Cioffi, "Sequence detection," *EE379B Course notes chapter 9. Stanford University*, 2008.

[24] L. Rokach, *Pattern classification using ensemble methods*. World Scientific, 2010, vol. 75.

[25] N. Shlezinger, R. Fu, and Y. C. Eldar, "DeepSIC: Deep soft interference cancellation for multiuser MIMO detection," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1349–1362, 2021.

[26] N. Shlezinger, J. Whang, Y. C. Eldar, and A. G. Dimakis, "Model-based deep learning," *arXiv preprint arXiv:2012.08405*, 2020.