Supplementary Materials

This is the supplementary materials section for "Constrained Predictive Coding as a Biologically Plausible Model of the Cortical Hierarchy".

A Derivation of upper bound

In this section we give a more detailed derivation of our framework, using a slightly modified order of steps compared to the main text. This should help clarify some aspects of the final algorithm.

Similar to the main text, we start with the predictive coding objective and introduce the weight doubles. We arrive at

$$
\min_{\mathbf{Z},\mathbf{W}} L = \min_{\mathbf{Z},\mathbf{W}_a,\mathbf{W}_b} \frac{1}{2} \sum_{l=1}^{n-1} \Big[g_b^{(l)} \big\| \mathbf{Z}^{(l)} - \mathbf{W}_b^{(l-1)} \mathbf{Z}^{(l-1)} \big\|_F^2
$$
$$
+ g_a^{(l+1)} \big\| \mathbf{Z}^{(l+1)} - \mathbf{W}_a^{(l)} \mathbf{Z}^{(l)} \big\|_F^2 + c^{(l)} \big\| \mathbf{Z}^{(l)} \mathbf{Z}^{(l)\top} \big\|_F^2 \Big], \quad (16)
$$

where we have also introduced a quadratic cost (a prior on $\mathbf{Z}$) with coefficient $c^{(l)}$. We now impose the constraint on the empirical covariance, $\frac{1}{T}\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top} \preceq \rho^{(l)}\mathbf{I}$, of the internal variables $\mathbf{Z}^{(l)}$[5]:

$$
\min_{\mathbf{Z},\mathbf{W}} L \leq \min_{\substack{\mathbf{Z},\mathbf{W}_a,\mathbf{W}_b \\ \frac{1}{T}\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top} \preceq \rho^{(l)}\mathbf{I}}} \frac{1}{2} \sum_{l=1}^{n-1} \Big[g_b^{(l)} \big\| \mathbf{Z}^{(l)} - \mathbf{W}_b^{(l-1)} \mathbf{Z}^{(l-1)} \big\|_F^2
$$
$$
+ g_a^{(l+1)} \big\| \mathbf{Z}^{(l+1)} - \mathbf{W}_a^{(l)} \mathbf{Z}^{(l)} \big\|_F^2 + c^{(l)} \big\| \mathbf{Z}^{(l)} \mathbf{Z}^{(l)\top} \big\|_F^2 \Big]. \quad (17)
$$

Expanding this and using the constraint we again upper bound the objective:

$$
\min_{\mathbf{Z},\mathbf{W}} L \leq \min_{\substack{\mathbf{Z},\mathbf{W}_a,\mathbf{W}_b \\ \frac{1}{T}\mathbf{Z}^{(l)}\mathbf{Z}^{(l)\top} \preceq \rho^{(l)}\mathbf{I}}} \frac{1}{2} \sum_{l=1}^{n-1} \Big[g_b^{(l)} \big\| \mathbf{Z}^{(l)} - \mathbf{W}_b^{(l-1)} \mathbf{Z}^{(l-1)} \big\|_F^2 + g_a^{(l+1)} \big\| \mathbf{Z}^{(l+1)} \mathbf{Z}^{(l+1)\top} \big\|_F^2
$$
$$
+ g_a^{(l+1)} \operatorname{Tr}\big(-2\mathbf{Z}^{(l+1)\top} \mathbf{W}_a^{(l)} \mathbf{Z}^{(l)} + T\rho^{(l)} \mathbf{W}_a^{(l)\top} \mathbf{W}_a^{(l)} \big) + c^{(l)} \big\| \mathbf{Z}^{(l)} \mathbf{Z}^{(l)\top} \big\|_F^2 \Big], \quad (18)
$$

We next implement the inequality constraint using positive-definite Lagrange multipliers:

$$
\min_{\mathbf{Z},\mathbf{W}} L \leq \min_{\mathbf{Z},\mathbf{W}_a,\mathbf{W}_b} \max_{\mathbf{Q}} \sum_{l=1}^{n-1} \tilde{L}^{(l)}, \quad (19)
$$

where

$$
\tilde{L}^{(l)} = \frac{1}{2} \Big[g_b^{(l)} \big\| \mathbf{Z}^{(l)} - \mathbf{W}_b^{(l-1)} \mathbf{Z}^{(l-1)} \big\|_F^2 + c_q^{(l)} \operatorname{Tr} \mathbf{Q}^{(l)\top} \mathbf{Q}^{(l)} \big(\mathbf{Z}^{(l)} \mathbf{Z}^{(l)\top} - T\rho^{(l)}\mathbf{I} \big)
$$
$$
+ g_a^{(l+1)} \operatorname{Tr}\big(-2\mathbf{Z}^{(l+1)\top} \mathbf{W}_a^{(l)} \mathbf{Z}^{(l)} + T\rho^{(l)} \mathbf{W}_a^{(l)\top} \mathbf{W}_a^{(l)} \big) \quad (20)
$$
$$
+ c^{(l)} \big\| \mathbf{Z}^{(l)} \mathbf{Z}^{(l)\top} \big\|_F^2 + g_a^{(l+1)} \big\| \mathbf{Z}^{(l+1)} \mathbf{Z}^{(l+1)\top} \big\|_F^2 \Big],
$$

where we have added a positive constant $c_q^{(l)}$ defining the coefficient of the inequality constraint. This constant does not affect the solution of the optimization problem but does affect the details of the neural dynamics. Motivated by biological arguments[6], we set this constant equal to $g_a^{(l)}$. In doing so, the interneuron variables no longer need a $(1/g_a^{(l)})$ in their definition, as in the main text, and are instead given by $\mathbf{n}^{(l)} = \mathbf{Q}^{(l)} \mathbf{z}^{(l)}$.

---

[5]The factors $\rho^{(l)}$ which set the scale of the constraint were set to 1 in the derivation in the main text for brevity. However, it is sensible for this scale factor to vary from layer to layer and possibly depend on the layer width. In Sec. C, we explore the effects of this parameter on the performance of the network.

[6]Since the feedback of the interneurons is received and integrated in the apical compartment, we expect the apical-to-soma conductance to also be present here.

1

A final upper bound to (19) can be found if each $\tilde{L}^{(l)}$ is optimized individually. Explicitly, we perform gradient descent of each variable, $\mathbf{Z}^{(l)}, \mathbf{W}_a^{(l)}, \mathbf{W}_b^{(l)}$, only with respect to $\tilde{L}^{(l)}$ and not with respect to the entire objective L:

$$
\begin{aligned}
\delta \mathbf{Z}^{(l)} &= \nabla_{\mathbf{Z}^{(l)}} L \;\rightarrow\; \nabla_{\mathbf{Z}^{(l)}} \tilde{L}^{(l)}, \\
\delta \mathbf{W}_a^{(l)} &= \nabla_{\mathbf{W}_a^{(l)}} L \;\rightarrow\; \nabla_{\mathbf{W}_a^{(l)}} \tilde{L}^{(l)}, \\
\delta \mathbf{W}_b^{(l)} &= \nabla_{\mathbf{W}_b^{(l)}} L \;\rightarrow\; \nabla_{\mathbf{W}_b^{(l)}} \tilde{L}^{(l)}.
\end{aligned}
\tag{21}
$$

This affords us an upper bound because we are no longer finding the actual minimum of L. Symbolically this can be written as in (9):

$$
\min_{\mathbf{Z},\mathbf{W}} L \le \sum_{l=1}^{n-1} \min_{\mathbf{Z},\mathbf{W}_a,\mathbf{W}_b} \max_{\mathbf{Q}} \tilde{L}^{(l)},
\tag{22}
$$

where the precise definition of this equation is given by the update equations (21). Note that these updates are no longer curl-free and hence cannot be found as the gradient of a single objective function.

B Theoretical section proofs

Note that the definition of ϵ in Proposition 1 is different from the definition in other sections.

B.1 Proof of Proposition 1

To prove Proposition 1, we first show that the latent variables $\mathbf{z}^{(l)}$ have a specific form. As in the main text, we define $\epsilon^{(l)} = g_a^{(l)}/g_b^{(l)}$ and, for simplicity, we assume $\epsilon^{(l)} = \epsilon$ is the same for all layers. For simplicity also, we assume $c^{(l)}$ (equivalently $d^{(l)}$ in previous section) is equal to zero, and use $c_q^{(l)} = g_a^{(l)}$ as in the previous section. This choice makes the notation simpler without changing our results. Equation (10) becomes

$$
\dot{\mathbf{z}}^{(l)} = g_b^{(l)} \mathbf{W}_b^{(l-1)} \mathbf{z}^{(l-1)} + g_a^{(l)} \mathbf{W}_a^{(l)\top} \mathbf{z}^{(l+1)} - g_b^{(l)} \mathbf{z}^{(l)} - g_a^{(l)} \mathbf{Q}^{(l)\top} \mathbf{Q}^{(l)} \mathbf{z}^{(l)}.
\tag{23}
$$

Proposition 2. *Assume the neural dynamics given in Eq. (23) are at equilibrium. Then the latent variables take the form*

$$
\mathbf{z}^{(n-k)} = \mathbf{B}_k \left(\mathbf{W}_b^{(n-k-1)} \mathbf{z}^{(n-k-1)} + \epsilon^k \mathbf{A}_k \mathbf{W}_a^{(n-1)\top} \mathbf{y} \right),
\tag{24}
$$

where $\mathbf{B}_k$ and $\mathbf{A}_k$ are independent of y and defined recursively via

$$
\begin{aligned}
\mathbf{A}_{k+1} &= \mathbf{W}_a^{(n-k-1)\top} \mathbf{B}_k \mathbf{A}_k, \\
\mathbf{B}_{k+1} &= \left(1 - \epsilon \mathbf{F}^{(n-k-1)} \mathbf{W}_a^{(n-k-1)\top} \mathbf{B}_k \mathbf{W}_b^{(n-k-1)} \right)^{-1} \mathbf{F}^{(n-k-1)},
\end{aligned}
$$

with $\mathbf{F}^{(l)} = \left(\mathbf{I} + \epsilon \mathbf{Q}^{(l)\top} \mathbf{Q}^{(l)} \right)^{-1}$ and

$$
\mathbf{B}_1 = \mathbf{F}^{(n-1)}, \quad \mathbf{A}_1 = 1.
$$

Proof. We prove this via induction. Starting from the neural dynamics Eq. (23), at equilibrium ($\dot{\mathbf{z}}^{(l)} = 0$) we have:

$$
\mathbf{z}^{(l)} = \mathbf{F}^{(l)} \left(\mathbf{W}_b^{(l-1)} \mathbf{z}^{(l-1)} + \epsilon \mathbf{W}_a^{(l)\top} \mathbf{z}^{(l+1)} \right).
$$

Setting $l = n - 1$ we get:

$$
\mathbf{z}^{(n-1)} = \mathbf{F}^{(n-1)} \left(\mathbf{W}_b^{(n-2)} \mathbf{z}^{(n-2)} + \epsilon \mathbf{W}_a^{(n-1)\top} \mathbf{y} \right),
$$

2

which satisfies the proposed form in Eq. (24) for $k = 1$. Now, we assume that the form in Eq. (24) holds for $z^{(n-k)}$, and show that it also holds for $z^{(n-k-1)}$. We have

$$\mathbf{z}^{(n-k-1)} = \mathbf{F}^{(n-k-1)} \left(\mathbf{W}_b^{(n-k-2)} \mathbf{z}^{(n-k-2)} + \epsilon \mathbf{W}_a^{(n-k-1)\top} \mathbf{z}^{(n-k)} \right)$$

$$= \mathbf{F}^{(n-k-1)} \left(\mathbf{W}_b^{(n-k-2)} \mathbf{z}^{(n-k-2)} + \epsilon \mathbf{W}_a^{(n-k-1)\top} \mathbf{B}_k \left(\mathbf{W}_b^{(n-k-1)} \mathbf{z}^{(n-k-1)} + \epsilon^k \mathbf{A}_k \mathbf{W}_a^{(n-1)\top} \mathbf{y} \right) \right).$$

Collecting $\mathbf{z}^{(n-k-1)}$ and multiplying by the inverse on the left we arrive at:

$$\mathbf{z}^{(n-k-1)} = \left(1 - \epsilon \mathbf{F}^{(n-k-1)} \mathbf{W}_a^{(n-k-1)\top} \mathbf{B}_k \mathbf{W}_b^{(n-k-1)} \right)^{-1} \mathbf{F}^{(n-k-1)} \times$$

$$\times \left(\mathbf{W}_b^{(n-k-2)} \mathbf{z}^{(n-k-2)} + \epsilon^{k+1} \mathbf{W}_a^{(n-k-1)\top} \mathbf{B}_k \mathbf{A}_k \mathbf{W}_a^{(n-1)\top} \mathbf{y} \right),$$

as proposed. □

Note that, whereas Proposition 1 holds only in the $\epsilon \to 0$ limit, Proposition 2 holds for general ϵ. Using this result we show that the relationship between the activity, feedback and feedforward inputs of the lth module has a simple form given in the following corollary:

Corollary 1. *The leading order in ϵ of the dependence of $\mathbf{z}^{(l)}$ and $\mathbf{z}^{(l+1)}$ on $\mathbf{z}^{(l-1)}$ and $\mathbf{y}$, which has non-zero dependence on $\mathbf{y}$, is given by*

$$\mathbf{z}^{(l)} = \mathbf{C}_1^{(l)} \mathbf{z}^{(l-1)} + \mathcal{O}(\epsilon^{n-l}),$$
$$\mathbf{z}^{(l+1)} = \mathbf{C}_2^{(l)} \mathbf{z}^{(l-1)} + \epsilon^{n-l-1} \mathbf{W}_a^{(l+1)\top} \cdots \mathbf{W}_a^{(n-1)\top} \mathbf{y} + \mathcal{O}(\epsilon^{n-l}),$$

where $\mathbf{C}_1^{(l)}$ and $\mathbf{C}_2^{(l)}$ are respectively the expansions of $\mathbf{B}_{n-l} \mathbf{W}_b^{(l-1)}$ and $\mathbf{B}_{n-l-1} \mathbf{W}_b^{(l)} \mathbf{B}_{n-l} \mathbf{W}_b^{(l-1)}$ up to order ϵ^{n-l-1}.

Proof. The result follows directly from applying Eq. (24) twice and then dropping all but the leading dependence on ϵ in the coefficient of the $\mathbf{y}$ term. □

Now, we restate and prove Proposition 1, which can be viewed as a generalization of the results in [55]. Note that in this proposition, the exact form of $\mathbf{C}_1^{(l)}$ and $\mathbf{C}_2^{(l)}$ is not important. All that matters is that both $\mathbf{z}^{(l)}$ and $\mathbf{z}^{(l+1)}$ have a linear dependence on $\mathbf{z}^{(l-1)}$. Since the dependence of $\mathbf{z}^{(l)}$ on $\mathbf{y}$ is subleading compared to the dependence of $\mathbf{z}^{(l+1)}$ on $\mathbf{y}$, this can be dropped in the limit of $\epsilon \to 0$.

Proposition 1. *Assume that the learning rules (Eqs. 11) are at equilibrium and we receive a new datapoint given by $\mathbf{x}_{T+1}, \mathbf{y}_{T+1}$. For $c^{(l)} = 0$ and in the limit of $\epsilon \equiv g_a/g_b \to 0$, the leading term in ϵ for the forward weight updates $\delta \mathbf{W}_b^{(l-1)}$ for this new sample is given by*

$$\delta \mathbf{W}_b^{(l-1)} \propto \mathbf{v}_{a,T+1}^{(l)} \mathbf{z}_{T+1}^{(l-1)\top} = \epsilon^{n-l} \left[\mathbf{W}_a^{(l)\top} \cdots \mathbf{W}_a^{(n-1)\top} (\mathbf{y}_{T+1} - \tilde{\mathbf{y}}_{T+1}) \right] \mathbf{z}_{T+1}^{(l-1)\top} + \mathcal{O}(\epsilon^{n-l+1}),$$

where $\tilde{\mathbf{y}}_{T+1} = \mathbf{Y} \mathbf{Z}^{(l-1)\top} \left(\mathbf{Z}^{(l-1)} \mathbf{Z}^{(l-1)\top} \right)^{-1} \mathbf{z}_{T+1}^{(l-1)}$ is the optimal linear inferred value for $\mathbf{y}_T$.

Proof. The statement follows from two applications of the update rule for $\mathbf{W}_b$ given in Eq. (11a) for $\mathbf{W}_b^{(l-1)}$. Putting $c^{(l)} = 0$ we have:

$$\delta \mathbf{W}_b^{(l-1)} \propto \epsilon \left(\mathbf{W}_a^{(l)\top} \mathbf{z}^{(l+1)} - \mathbf{Q}^{(l)\top} \mathbf{Q}^{(l)} \mathbf{z}^{(l)} \right) \mathbf{z}^{(l-1)\top}. \tag{25}$$

The equilibrium condition given by $\delta \mathbf{W}_b = 0$ is given by averaging over the entire dataset:

$$\mathbf{Q}^{(l)\top} \mathbf{Q}^{(l)} \mathbf{Z}^{(l)} \mathbf{Z}^{(l-1)\top} = \mathbf{W}_a^{(l)\top} \mathbf{Z}^{(l+1)} \mathbf{Z}^{(l-1)\top}. \tag{26}$$

Now we plug the derived values in Corollary 1. After right-multiplying by $(\mathbf{Z}^{(l-1)} \mathbf{Z}^{(l-1)\top})^{-1}$ we get:

$$\mathbf{Q}^{(l)\top} \mathbf{Q}^{(l)} \mathbf{C}_1 = \mathbf{W}_a^{(l)\top} \mathbf{C}_2 + \epsilon^{n-l-1} \mathbf{W}_a^{(l)\top} \cdots \mathbf{W}_a^{(n-1)\top} \mathbf{Y} \mathbf{Z}^{(l-1)\top} (\mathbf{Z}^{(l-1)} \mathbf{Z}^{(l-1)\top})^{-1}. \tag{27}$$

3

Now, we look at the update for a new sample $T + 1$. Plugging the above as well as the equations from Corollary 1 into the update rule we see that the $\mathbf{C}_2^{(l)}$ dependent term cancels and we are only left with the term depending on $\mathbf{y}$:

$$
\begin{aligned}
\delta\mathbf{W}_b^{(l-1)} &\propto \epsilon^{n-l}\left(\mathbf{W}_a^{(l)\top}\mathbf{z}_{T+1}^{(l+1)} - \mathbf{Q}^{(l)\top}\mathbf{Q}^{(l)}\mathbf{z}_{T+1}^{(l)}\right)\mathbf{z}_{T+1}^{(l-1)\top} \\
&= \epsilon^{n-l}\left[\mathbf{W}_a^{(l)\top}\cdots\mathbf{W}_a^{(n-1)\top}(\mathbf{y}_{T+1} - \tilde{\mathbf{y}}_{T+1})\right]\mathbf{z}^{(l-1)\top},
\end{aligned}
$$

with $\tilde{\mathbf{y}}_{T+1}$ as defined in the proposition. □

B.2 Proof of Theorem 1 and consequences

In this section we look at some properties of the within-layer computations of CCPC. We will focus on the computations within the lth layer and for clarity of notation, for this subsection only, we rename $\mathbf{Z}^{(l)} \to \mathbf{Z}$, $\mathbf{Z}^{(l+1)} \to \mathbf{Y}$, and $\mathbf{Z}^{(l-1)} \to \mathbf{X}$. We will assume $\rho^{(l)} = 1$ for brevity. The results have straight-forward generalizations for the case where $\rho^{(l)} \neq 1$.

Theorem 1. *Let the concatenated matrix $\mathbf{\Xi} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^\top = [\epsilon^{1/2}\mathbf{C}_x^{-1/2}\mathbf{X}, \mathbf{Y}] \in \mathbb{R}^{(d_X+d_Y)\times T}$ have singular value decomposition $\mathbf{\Xi} = \sum_{\alpha=1}^{(d_X+d_Y)}\mathbf{u}_\alpha\lambda_\alpha\mathbf{v}_\alpha^\top$, with $\{\mathbf{u}_\alpha \in \mathbb{R}^{(d_X+d_Y)}\}$ and $\{\mathbf{v}_\alpha \in \mathbb{R}^T\}$ both being sets of orthonormal vectors, and singular values (λ_α) sorted in decreasing order. We consider the minimization*

$$
\min_{\substack{\mathbf{Z}\in\mathbb{R}^{d\times N} \\ \mathbf{Z}\mathbf{Z}^\top\preccurlyeq I_d}} \text{Tr}\left[(c+\epsilon)\mathbf{Z}^\top\mathbf{Z} - \mathbf{Z}^\top\mathbf{Z}[\epsilon\mathbf{X}^\top(\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X} + \mathbf{Y}^\top\mathbf{Y}]\right]. \tag{28}
$$

Then one of the optimal solutions is given by $\hat{\mathbf{Z}} = \sum_{\alpha=1}^{D}\mathbf{w}_\alpha\mathbb{1}(\lambda_\alpha > c+\epsilon)\mathbf{v}_\alpha^\top$ where $\{\mathbf{w}_\alpha \in \mathbb{R}^d\}$ is an arbitrary set of orthonormal vectors, with $D = \min(d, d_X + d_Y)$.

Proof. Let the SVD of $\mathbf{Z}$ be $\mathbf{W}\mathbf{\Lambda}_{\mathbf{Z}}\mathbf{V}_{\mathbf{Z}}^\top$. Minimizing over $\mathbf{Z}$ amounts to finding the matrices $\mathbf{W}$, $\mathbf{\Lambda}_{\mathbf{Z}}$, and $\mathbf{V}_{\mathbf{Z}}$ that give the smallest value of the objective. Because of the constraint $\mathbf{Z}\mathbf{Z}^\top \preccurlyeq I_d$, $\mathbf{\Lambda}_{\mathbf{Z}}$ satisfies $0 \leq \lambda_{\mathbf{Z}\alpha} \leq 1$. Plugging this form into the optimization objective we have

$$
\text{Tr}\left[(c+\epsilon)\mathbf{Z}^\top\mathbf{Z} - \mathbf{Z}^\top\mathbf{Z}[\epsilon\mathbf{X}^\top(\mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X} + \mathbf{Y}^\top\mathbf{Y}]\right] = \text{Tr}\left[(c+\epsilon)\mathbf{\Lambda}_{\mathbf{Z}}^\top\mathbf{\Lambda}_{\mathbf{Z}} - \mathbf{\Lambda}_{\mathbf{Z}}^\top\mathbf{\Lambda}_{\mathbf{Z}}\tilde{\mathbf{V}}^\top\mathbf{\Lambda}^\top\mathbf{\Lambda}\tilde{\mathbf{V}}\right]
$$

where $\tilde{\mathbf{V}} = \mathbf{V}^\top\mathbf{V}_{\mathbf{Z}}$. Note that $\mathbf{W}$ does not appear in this expression and therefore remains undetermined in this optimization objective.

$\mathbf{V}$ only appears in the second term and hence can be found by optimizing the second term alone. This term is of the form $-\sum_{\alpha,\beta}\lambda_{\mathbf{Z}\alpha}^2|\tilde{V}_{\alpha\beta}|^2\lambda_\beta^2$. The $\tilde{\mathbf{V}}$ that minimizes the term is the one that pairs λ_β and $\lambda_{\mathbf{Z}\alpha}$, in the sorted order. Assuming $(\lambda_{\mathbf{Z}\alpha})$ are also sorted in the decreasing order, this is achieved by setting $\mathbf{V}_{\mathbf{Z}} = \mathbf{V}$.

Plugging this in, the entire expression simplifies to

$$
-\sum_\alpha(\lambda_\alpha - c - \epsilon)\lambda_{\mathbf{Z}\alpha}.
$$

Since $0 \leq \lambda_{\mathbf{Z}\alpha} \leq 1$, it is clear that, for a minimum, if $(\lambda_\alpha - c - \epsilon) < 0$, $\lambda_{\mathbf{Z}\alpha} = 0$, while $(\lambda_\alpha - c - \epsilon) > 0$ implies $\lambda_{\mathbf{Z}\alpha} = 1$. In case of equality, the choice of $\lambda_{\mathbf{Z}\alpha}$ does not matter. Hence $\lambda_{\mathbf{Z}\alpha} = \mathbb{1}(\lambda_\alpha > c + \epsilon)$ is a valid minimum. The vectors $\{\mathbf{w}_\alpha\}$ are the columns of $\mathbf{W}$, while $\{\mathbf{v}_\alpha\}$ are the columns of $\mathbf{V}$. Hence, the result. □

From this theorem, we derive two corollaries, describing the different limits of $\epsilon \to 0$ and $\epsilon \to \infty$. The proofs of these corollaries follow directly from Theorem 1 which maps the within-layer computation of CCPC onto a PCA problem, and the known properties of PCA.

Corollary 2. *If the input sample covariance matrix, $\mathbf{C}_x$, is full rank, the internal representation $\mathbf{Z}$ in the limit $\epsilon \to 0$ is driven by the principal components (scores) of $\mathbf{Y}$, i.e.,*

$$
\mathbf{Z}^* = \mathbf{W}_a^\top\mathbf{Y} + \epsilon\mathbf{W}_b\mathbf{X}, \tag{29}
$$

with $\mathbf{W}_a$ the top principal directions of $\mathbf{Y}$ and $\mathbf{W}_b$ the maximally correlated directions between $\mathbf{X}$ and the columns of $\mathbf{W}_a$.

4

Corollary 3. *If the input sample covariance matrix, $\mathbf{C}_x$, is full rank, the internal representation $\mathbf{Z}$ in the limit $\epsilon \to \infty$ is driven by the maximally correlated directions between $\mathbf{C}_x^{-1/2}\mathbf{X}$ and $\mathbf{Y}$, i.e.,*

$$\mathbf{Z}^* = \mathbf{W}_a^\top \mathbf{Y} + \mathbf{W}_b \mathbf{X}, \tag{30}$$

with $\mathbf{W}_a, \mathbf{W}_b$ projecting onto a common lower-dimensional subspace so that the projections are maximally correlated.

B.3 Rank structure of CCPC

In this section again we set $c^{(l)} = 0$ and $\rho^{(l)} = 1$, and assume $g_a = 1, g_b = \epsilon$ is the same for all layers. The generalization to other cases is straightforward. We demonstrate a series of rank constraints on the intermediate layers of CCPC. The first of these is very general and requires no assumptions. The following statements become gradually more tight but also require more assumptions.

In this section, we define r_x and r_y to be the ranks of the the covariances of $\mathbf{X}$ and $\mathbf{Y}$, and $r^{(l)}$ to be the rank of the covariance of $\mathbf{Z}^{(l)}$ at the equilibrium of the CCPC update equations. We take $r^{(0)} = r_x$ and $r^{(n)} = r_y$.

Corollary 4. *The rank of the lth layer is bounded by $r^{(l+1)} + r^{(l-1)}$:*

$$\forall l \; : \; r^{(l)} \leq r^{(l+1)} + r^{(l-1)}. \tag{31}$$

Proof. This follows directly from Theorem 1 which describes $Z^{(l)}$ as the whitened PCA of the concatenation of $\mathbf{Z}^{(l-1)}$ and $\mathbf{Z}^{(l+1)}$. □

When the adjacent layers are whitened, i.e., the non-zero eigenvalues of $Z^{(l+1)}Z^{(l+1)\top}$ and $Z^{(l-1)}Z^{(l-1)\top}$ are all one, we can prove a tighter form of the rank constraint. This is relevant for the intermediate layers of CCPC other than the final hidden layer. This proposition is relevant for the final hidden layer only if the supervisory input $\mathbf{Y}$ is white.

Proposition 3. *Let $r_c^{(l)}$ be the rank of the covariance matrix $\mathbf{Z}^{(l-1)}\mathbf{Z}^{(l+1)\top}$. The rank of intermediate layers $l \leq n - 2$ is given by:*

$$\forall l \leq n - 2 \; : \; r^{(l)} \leq \begin{cases} r_c^{(l)} \leq \min\{r^{(l+1)}, r^{(l-1)}\} & \epsilon \geq 1, \\ r^{(l+1)} & \epsilon < 1. \end{cases} \tag{32}$$

Proof. This is a slight generalization of [74]. It follows from the eigenvalue structure of the concatenated data matrix given in Theorem 1, where only a subset of the eigenvalues exceed the threshold given by ϵ. The number of eigenvalues that exceed this threshold is given by Eq. (32). □

Applying this proposition to all the layers of a CCPC network, we arrive at the following corollary:

Corollary 5. *Assume that $\mathbf{Y}$ is white and $\epsilon \geq 1$. In a CCPC network with $c^{(l)} \geq 0$, $\rho^{(l)} = 1$, and layer widths $w^{(l)}$, the rank of the latent variables $\mathbf{z}^{(l)}$ is less than or equal to $\min\{r_x, r_y, w^{(1)}, \cdots, , w^{(n-1)}\}$.*

In other words, the rank of the latent variables is bounded by the minimum of the input/output ranks and the bottleneck of the network.

B.4 Alignment and relation to CCA

In the limit where the internal loss terms in Eq. (6) are zero, we effectively have a deep linear network [79]. A lot is known about deep linear networks, trained by gradient descent of weights. One of the remarkable properties of such networks is that the weights remain balanced, namely $\mathbf{W}^{(l)\top}\mathbf{W}^{(l)} - \mathbf{W}^{(l-1)}\mathbf{W}^{(l-1)\top}$ remains a constant, under gradient descent. In the case this difference is zero, the left singular vectors of $\mathbf{W}^{(l-1)}$ match the right singular vectors of $\mathbf{W}^{(l)}$ [83]. In addition, for two class linear discriminants, $\mathbf{W}^{(l-1)}$ ultimately become rank 1, under gradient descent [83].

5
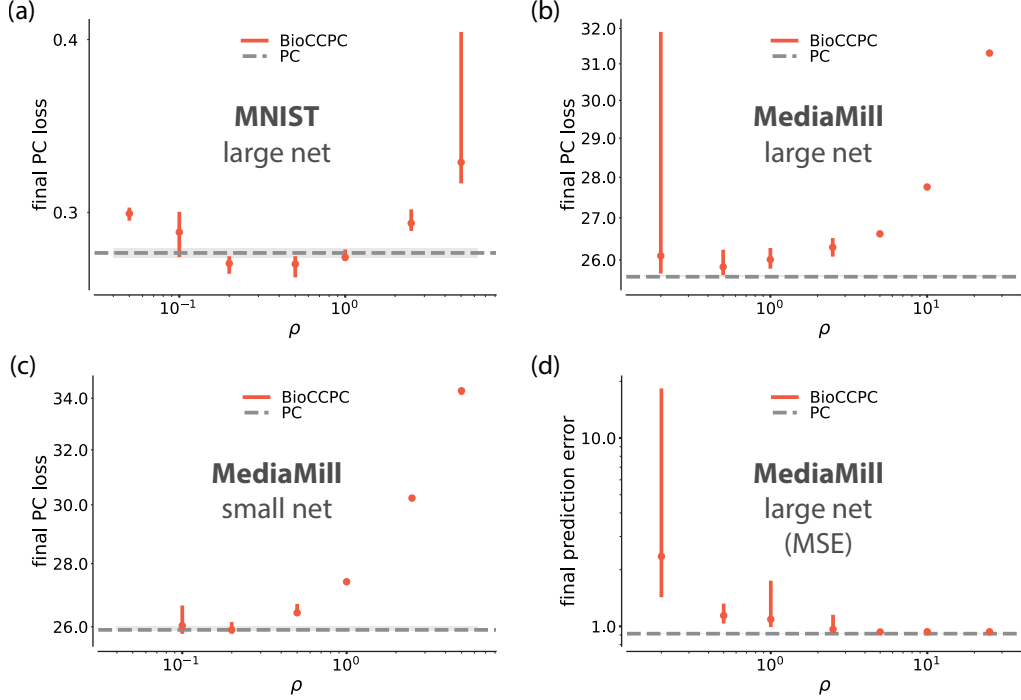
Figure 4: Dependence of BioCCPC performance on constraint scale. Unless otherwise noted, this is the predictive-coding loss (Eq. 6) after 500,000 iterations. (a) A large network with two hidden layers of sizes 50 and 5 trained on MNIST. (b) The same large network as in (a) run on the Mediamill dataset. See text. (c) A small net (single hidden layer of size 5) on Mediamill. (d) Mean-squared prediction error for the large net on Mediamill, showing that the best prediction performance may occur at a different choice of ρ compared to the lowest predictive-coding loss.

C Numerical simulations

C.1 Effect of constraint scale

The BioCCPC network uses an inequality constraint on the covariance of the hidden variables, $\frac{1}{T}\mathbf{Z}^{(l)}Z^{(l)\top} \preceq \rho^{(l)}\mathbf{I}$, and this constraint is saturated at convergence.[7] The scale, $\rho^{(l)}$, is therefore important: a scale that is too small or too big forces the hidden-layer activities into a range that can hinder learning, as seen in Fig. 4a. Interestingly, the BioCCPC algorithm with a well-chosen constraint scale converges faster than a plain PC network, that is, it can reach a lower PC loss given the same number of training samples (Fig. 4a).

The predictive-coding loss, Eq. (6), penalizes prediction errors in all layers. If we think about the task as supervised learning, however, it might make more sense to focus only on the accuracy of the prediction in the last layer. The best output prediction error may require a different constraint scale compared to the lowest PC loss, cf. Fig. 4b and Fig. 4d.

C.2 Effect of dataset choice

While we derived our algorithm by starting from the supervised PC framework of [4], the same derivation would hold for the unsupervised or self-supervised PC networks. In these settings, PC can be thought of as a framework which combines information from the inputs at the first and last layer, rather than a method for predicting an output from an input. In the linear case, this has much in common with biologically plausible circuits that perform canonical correlation

---

[7]More precisely, the constraint is saturated only within the subspace that $\mathbf{Z}^{(l)}$ occupies. In general, this is only the case in the layer with the smallest dimension, as the activity in the other layers is restricted to a subspace of dimension given by the smallest layer; see Corollary 5.

6
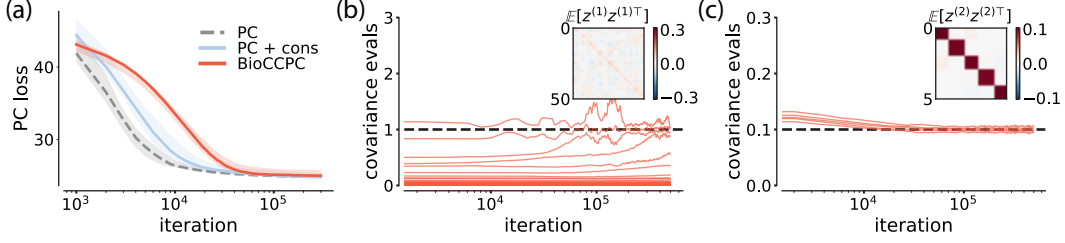
Figure 5: Comparison of our algorithm with PC on the Mediamill dataset. Simulations run using a network with two hidden layers, sizes 50 and 5. Shaded area represents the variation over 10 runs. (a) Evolution of the predictive-coding loss, Eq. (6), on a validation set during learning. (b) and (c) Evolution of the eigenvalues of the covariance matrix of hidden-layer activations in the two hidden layers. The dashed lines show the scale of the constraint; see SM Sec. A. Inset: covariance matrix at end of training. Note how activity is whitened in the second hidden layer (c), and low-rank in the first hidden layer (b). Note also that the network performance is good even though the constraint is not yet fully converged.

analysis (CCA) [55, 74]. We therefore tested the algorithm on Mediamill [84], a dataset consisting of two views of the same scene, one given by the video and the other by text annotation. The results (Figs. 4b-d and Fig. 5) show similar trends as those obtained on MNIST. In particular, an appropriate choice of constraint scale leads to results that are competitive with less biologically plausible implementations of PC.

In Fig. 6 we show the evolution of the validation-set loss for four other datasets: FashionMNIST, CIFAR10, CIFAR100, and the LFW (Labeled Faces in the Wild). The first three are supervised-learning scenarios and we use a network similar to the one we employed for MNIST. The LFW dataset contains pairs of views of the same subject, and so we employ a symmetric autoencoder-like architecture for this self-supervised task. As before, we see that the BioCCPC algorithm converges to a solution that has predictive-coding loss comparable to the minimum obtained from traditional PC implementations (see Fig. 6).
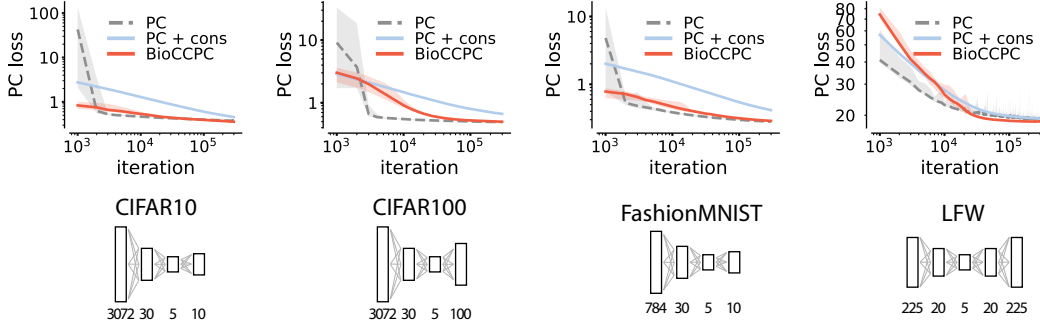
Figure 6: Comparison of our algorithm with PC on a variety of datasets. The name of the dataset and the architecture we employed are shown below each figure. Shaded area represents the variation over 10 runs.

C.3 Network architecture

We tested BioCCPC across different architectures, such as a shallow network with a small hidden-layer dimension (Fig. 4) and a deeper network with two hidden layers, one of which is large-dimensional (Fig. 4a,b,d). In both cases BioCCPC compares well to PC.

C.4 Lack of symmetry in feedforward and feedback weights

One of the difficulties with a biological implementation of predictive coding is that it requires symmetric weights across layers. This is not so much an issue of plausibility—indeed, symmetric weights can be obtained simply by using the same Hebbian rules to train both feedforward and feedback connections—but rather a problem with realism: weights in many areas of the brain are not in fact seen to be symmetric.

7
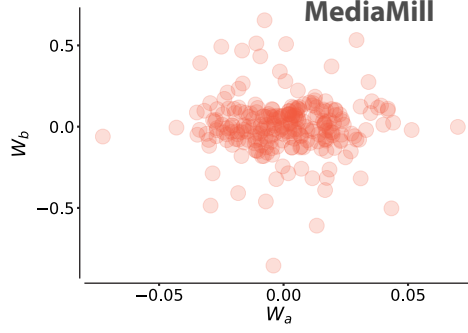
Figure 7: Comparison of feedforward and feedback weights after learning on a network with hidden layers of sizes 50 and 5 trained on the Mediamill dataset.
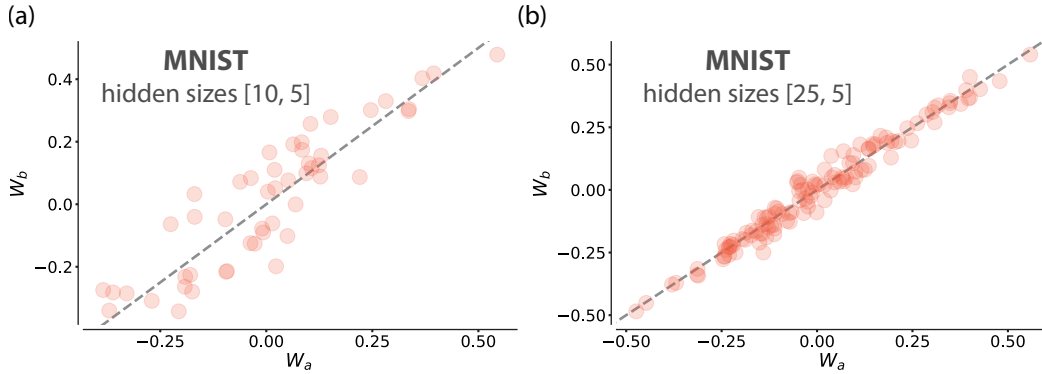
Figure 8: Comparison of feedforward and feedback weights after learning for different network sizes. This is trained on MNIST. (a) There is less symmetry in a smaller network with hidden layers of sizes 10 and 5. (b) There is more symmetry in a larger network, with hidden layers 25 and 5.

Our algorithm explicitly breaks the symmetry between the weights, leading to solutions that generically alleviate this difficulty. Indeed, in some cases there is very little correlation between feedforward and feedback weights; see Fig. 7. However, in some cases we see that the forward and backward weights converge to similar values even though the learning rules are not symmetric. We suspect that this can occur in cases where there is a large imbalance between the width of adjacent layers, the larger the imbalance (i.e. the lower rank the wider layer is) the more correlation we see in our experiments; see Fig. 8. In biologically relevant neural networks, we do not generally see this large imbalance in the width of adjacent layers and therefore we do not expect to see high correlation between the forward and backward weights.

C.5 Predictive coding with constraint

In the main text we considered a predictive-coding network that imposes a covariance constrained but we did not make explicit what this entails. Explicitly, we augmented the Whittington-Bogacz objective, Eq. (2), with a Lagrange multiplier enforcing an inequality constraint on the covariance of hidden variables, similar to what we do in BioCCPC:

$$L_{\text{cons}} = \frac{1}{2} \sum_l \frac{\left\| \mathbf{Z}^{(l)} - \mathbf{W}^{(l-1)} \mathbf{Z}^{(l-1)} \right\|_F^2}{\sigma^{(l)2}} + \text{Tr}\, \mathbf{Q}^{(l)\top} \mathbf{Q}^{(l)} (\mathbf{Z}^{(l)} \mathbf{Z}^{(l)\top} - T\rho^{(l)} \mathbf{I}), \qquad (33)$$

and optimized using stochastic gradient descent. This lets us distinguish the effect of the constraint from the effect of the approximations we used in the derivation.

8

C.6 Details of simulation runs

We initialized our the weights in our networks using uniform random numbers in the range $[-a, a]$, where $a = 1/\sqrt{\text{number of columns}}$. This is similar to Xavier initialization [85] except the scale of initial weights depends only on the number of columns instead of both rows and columns. We have found this initialization to provide slightly better results.

We optimized hyperparameters using Optuna [86] with the tree-structured Parzen Estimator (TPE) algorithm [87]. We ran the optimization for 100 trials and repeated it 10 times with different random seeds for every combination of dataset, network architecture, and constraint scale ρ (with the exception of the plain PC algorithm which does not have a constraint). In each case we optimized the learning rate η used in the weight updates and the learning rate τ^{-1} of the fast dynamics (see Algorithm 1). For networks that have a constraint, we allowed a different learning rate $f\eta$ for the $\mathbf{Q}$ dynamics, and optimized for the factor f. Finally, for BioCCPC network, we allowed for different learning rates for the forward and backward weights, $\mathbf{W}_b$ and $\mathbf{W}_a$, and optimized for the ratio between these learning rates. Hyperparameter optimization runs were trained for 500 batches with batch size 100 and were evaluated on a held-out validation set comprising 500 samples. Each run was repeated 4 times with different random initialization to account for stochasticity in the learning dynamics.

The simulation runs that were used to make the figures in the paper were based on the hyperparameter optimization results that yielded the lowest predictive-coding loss. Since hyperparameter optimization tends to push the learning rate up to the brink of instability, the learning rate used in the simulations was lowered by multiplying by a certain "safety" factor α. We generally used $\alpha = 0.8$.

We ran our simulations on the CPU. Each run took from 1 to 4 minutes when run single-threaded on a 24-core Intel Xeon CPU E5-2643 at 3.40GHz running Linux. We also used a cluster to run the hyperparameter optimization for different architectures in parallel.

D Generalization error
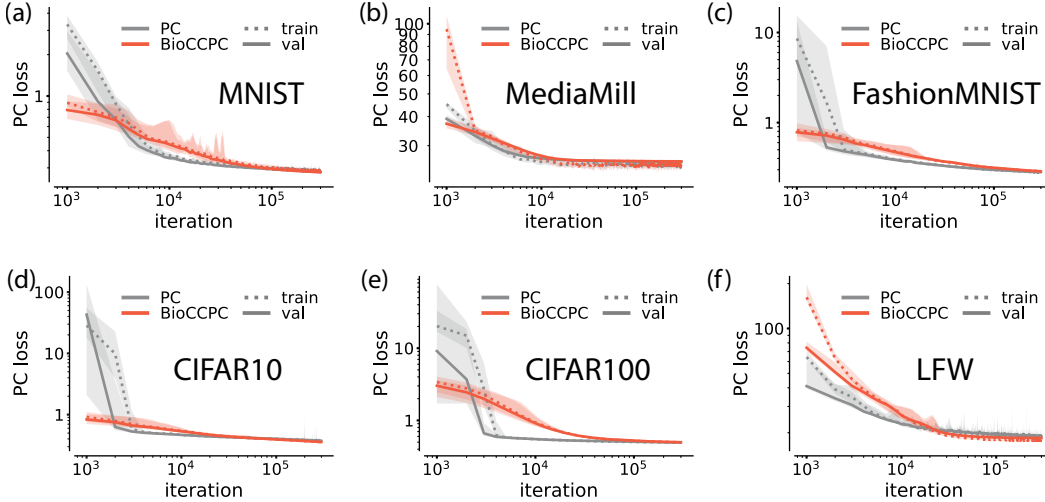
Figure 9: There is little overfitting in the linear setting that we are investigating. The plots compare the training- and validation-set predictive-coding losses for both the less biologically plausible predictive-coding implementation from [4] (PC) and for our algorithm (BioCCPC).

The linear constraint that we impose on our network is a strong regularizer. Figure 9 compares the training and validation losses during training on six different datasets (as described above). We see that there is very little overfitting in all cases.

9

E BioCCPC Pseudocode

Algorithm 1: Covariance Constrained predictive coding algorithm (BioCCPC)

---

input: $\mathbf{x}_t, \mathbf{y}_t$ ▷ new sample and previous weight matrices

    $\mathbf{W}_a^{(l)}, \mathbf{W}_b^{(l-1)}, \mathbf{Q}^{(l)}$ $\forall l \in \{1, \dots, n-1\}$

$\mathbf{z}_t^{(0)} \leftarrow \mathbf{x}_t,\ \mathbf{z}_t^{(l)} \leftarrow \mathbf{W}_b^{(l-1)}\mathbf{z}_t^{(l-1)}$ ▷ initialize latents via forward pass

run until convergence for all $l \in \{1, n-1\}$ ▷ neural dynamics

    $\mathbf{n}^{(l)} \leftarrow (1/g_a^{(l)})\mathbf{Q}^{(l)}\mathbf{z}_t^{(l)}$

    $\mathbf{v}_a^{(l)} \leftarrow \mathbf{W}_a^{(l)\top}\mathbf{z}_t^{(l+1)} - \mathbf{Q}^{(l)\top}\mathbf{n}^{(l)}$, $\mathbf{v}_b^{(l)} \leftarrow \mathbf{W}_b^{(l-1)}\mathbf{z}_t^{(l-1)}$

    $\mathbf{z}_t^{(l)} \leftarrow \mathbf{z}_t^{(l)} + \tau^{-1}\left[-g_{\text{lk}}^{(l)}\mathbf{z}_t^{(l)} + g_a^{(l)}(\mathbf{v}_a^{(l)} - \mathbf{z}_t^{(l)}) + g_b^{(l)}(\mathbf{v}_b^{(l)} - \mathbf{z}_t^{(l)})\right]$

update for all $l \in \{1, n-1\}$ ▷ synaptic weight updates

    $\mathbf{W}_a^{(l)} \leftarrow \mathbf{W}_a^{(l)} + \eta\big(\mathbf{z}_t^{(l+1)}\mathbf{z}_t^{(l)\top} - \mathbf{W}_a^{(l)}\big)$

    $\mathbf{W}_b^{(l-1)} \leftarrow \mathbf{W}_b^{(l-1)} + \eta\Big[g_a^{(l)}\big(\mathbf{W}_a^{(l)\top}\mathbf{z}_t^{(l+1)} - \mathbf{Q}^{(l)\top}\mathbf{n}^{(l)}\big) - c^{(l)}\mathbf{z}_t^{(l)}\Big]\mathbf{z}_t^{(l-1)\top}$

    $\mathbf{Q}^{(l)} \leftarrow \mathbf{Q}^{(l)} + \eta\big(\mathbf{n}^{(l)}\mathbf{z}_t^{(l)\top} - \mathbf{Q}^{(l)}\big)$

output: $\mathbf{z}_t^{(l)}$ $\forall l \in \{0, \dots, n\}$ ▷ internal representations and updated weights

    $\mathbf{W}_a^{(l)}, \mathbf{W}_b^{(l-1)}, \mathbf{Q}^{(l)}$ $\forall l \in \{1, \dots, n-1\}$

Supplementary References

[83] Ziwei Ji and Matus Telgarsky. Gradient descent aligns the layers of deep linear networks. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

[84] Cees GM Snoek, Marcel Worring, Jan C Van Gemert, Jan-Mark Geusebroek, and Arnold WM Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *Proceedings of the 14th ACM international conference on Multimedia*, pages 421–430, 2006.

[85] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

[86] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[87] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyperparameter optimization. *Advances in neural information processing systems*, 24, 2011.

10