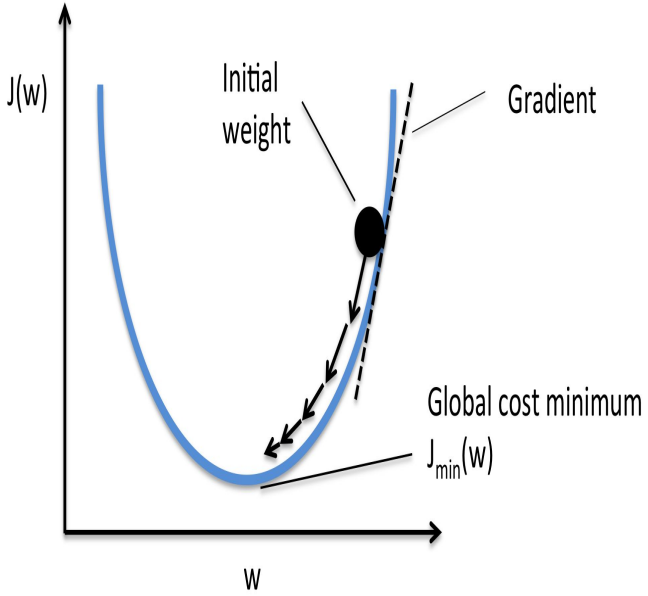
# Learning Overparameterized Neural Networks via Stochastic Gradient Descent on Structured Data

Yuanzhi Li, Yingyu Liang

Harshavardhan Reddy Bommireddy
Pranaya Satwika Reddy Maddi

# Stochastic Gradient Descent

- A method to find optimal parameter configuration
- Iteratively makes small adjustments to the network to decrease the error of the network
- Makes our model learn a lot faster even with large datasets



# Overparameterization in Neural Networks

- **Overparameterization**: number of model parameters exceed the size of training dataset
- Usually these tend to overfit
- But, it is empirically observed that learning with stochastic gradient descent in the overparameterized setting does not lead to overfitting

# Challenges with Existing work

- Recent studies use low complexity of the learned solution to explain the generalization
- Usually do not explain how SGD favours low complexity solutions
- **Observations:**
  - Overparameterization and proper random initialization helps optimization
  - Not understood why a particular initialization can help the optimization
  - Unrealistic assumption about data
    - Eg: Gaussian-ness or linear separability

# In this paper

- Learning a two-layer overparameterized neural network using SGD for classification
- More realistic structure of the data
  - Data in each class is mixture of several components
  - Components from different classes are well separated in distance
  - Components within each class can be close to each other
  - Eg: In MNIST dataset, each class corresponds to a digit and a class can have several components which correspond to different writing styles of the digit

# Contd...

- Through this paper, it's proved that when the network is sufficiently overparameterized, SGD probably learns a network close to random initialization with a small generalization error.
- Also shows that in a overparameterized setting, though the network can overfit, SGD with random initialization leads to good generalization
- Results shows that learning time depends on the parameters but not on the dimension of the data
- Success of learning relies on overparameterization and random initialization

# Related Work

- **Generalization of Neural Networks:**
  - Practical neural network have good generalization when trained on practical data
  - Good generalization of overparameterized network cannot be explained by traditional theory
  - Existing works do not address why there is low complexity

# Contd

- **Overparameterization and implicit regularization:**
  - Learning a two-layer overparameterized network on linearly separable data shows that SGD converges to a global optimum with good generalization
- **Theoretical analysis of learning neural networks:**
  - There exists lot of work that analyses the optimization of learning neural networks, but they assume unrealistic assumptions about the data

# Problem Setup

K-classes classification with two layer neural network with ReLU activation

$$f_i(x) = \sum_{r=1}^{m} a_{i,r} \mathbf{ReLU}(\langle w_r, x \rangle)$$

Where $w_r$ are the weights for m neurons and $a_{i,r}$ are the weights of the top layer

**ReLU(z) = max{0, z}**

# Assumptions about the Data

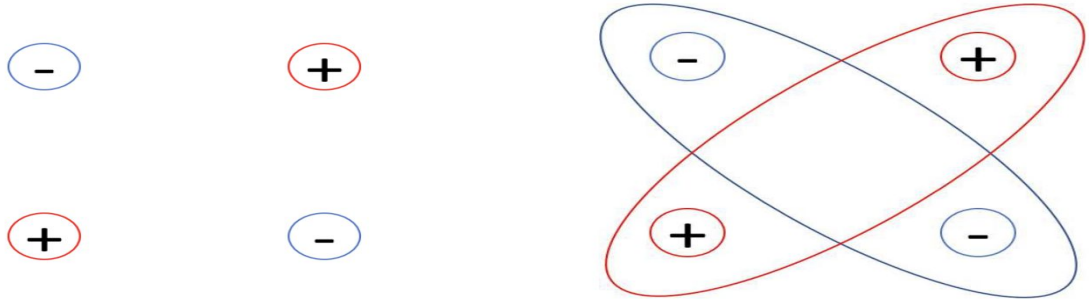
Data is generated from a Distribution D. There are kxl unknown distributions {D}

**(A1)** (Separability) There exists $\delta > 0$ such that for every $i_1 \neq i_2 \in [k]$ and every $j_1, j_2 \in [l]$, $\text{dist}\left(\text{supp}(\mathcal{D}_{i_1,j_1}), \text{supp}(\mathcal{D}_{i_2,j_2})\right) \geq \delta$.Moreover, for every $i \in [k], j \in [l]$,[1] $\text{diam}(\text{supp}(\mathcal{D}_{i,j})) \leq \lambda\delta$, for $\lambda \leq 1/(8l)$.

**(A2)** (Normalization) Any $x$ from the distribution has $\|x\|_2 = 1$.

we allow an arbitrary l ≥ 1 distributions in each class

# Illustration of the Separability Assumption




(a) Each class as two components (b) Each class as one component

Figure 3: Illustration of the separability assumption. The data lie in $\mathcal{R}^2$ and are from two classes $-$ and $+$. The $+$ class contains points uniformly over two balls of diameter $1/10$ with centers $(0, 0)$ and $(2, 2)$, and the $-$ class contains points uniformly over two balls of the same diameter with centers $(0, 2)$ and $(2, 0)$. (a) We can view each ball in each class as one component, then the data will satisfy the separability assumption with $\ell = 2$. (b) We can also view each class as just one component, but the data will not satisfy the separability assumption with $\ell = 1$.

Data satisfies separability assumption with l=2, but not when l=1

This shows allowing l>=2 leads to more flexibility, the assumption captures non linear structure of practical data better than linear separability

# Assumptions about Learning Process

We assume learning is from a random initialization

**(A3)** (Random initialization) $w_r^{(0)} \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$, $a_{i,r} \sim \mathcal{N}(0,1)$, with $\sigma = \frac{1}{m^{1/2}}$.

Cross entropy loss over softmax is defined as:

$$L(w) = -\frac{1}{N} \sum_{s=1}^{N} \log o_{y_s}(x_s, w), \text{ where } o_y(x, w) = \frac{e^{f_y(x,w)}}{\sum_{i=1}^{k} e^{f_i(x,w)}}.$$

# Contd

minibatch SGD of batch size B, number of iterations T = N/B and learning rate η

We randomly divide total training examples into T batches each of size B

Update at each iteration: $$w_r^{(t+1)} = w_r^{(t)} - \eta \frac{1}{B} \sum_{s \in \mathcal{B}_t} \frac{\partial L(w^{(t)}, x_s, y_s)}{\partial w_r^{(t)}}, \forall r \in [m]$$

where $$\frac{\partial L(w, x_s, y_s)}{\partial w_r} = \left( \sum_{i \neq y_s} a_{i,r} o_i(x_s, w) - \sum_{i \neq y_s} a_{y_s,r} o_i(x_s, w) \right) 1_{\langle w_r, x_s \rangle \geq 0} x_s$$

# Result

**Theorem 4.1.** *Suppose the assumptions (A1)(A2)(A3) are satisfied. Then for every* $\varepsilon > 0$, *there is* $M = poly(k, l, 1/\delta, 1/\varepsilon)$ *such that for every* $m \geq M$, *after doing a minibatch SGD with batch size* $B = poly(k, l, 1/\delta, 1/\varepsilon, \log m)$ *and learning rate* $\eta = \frac{1}{m \cdot poly(k,l,1/\delta,1/\varepsilon,\log m)}$ *for* $T = poly(k, l, 1/\delta, 1/\varepsilon, \log m)$ *iterations, with high probability:*

$$\Pr_{(x,y)\sim\mathcal{D}}\left[\forall j \in [k], j \neq y, f_y(x, w^{(T)}) > f_j(x, w^{(T)})\right] \geq 1 - \varepsilon.$$

- Total number of iterations can only be increased by factor of log*m*
- We can over parameterize the network without significantly increasing the complexity

# Analysis of the Theorem

- We can treat each example as a single distribution, implying λ is always zero
- We use batch size B for T iterations, Hence l = N = BT
- Input data is actually structured,  SGD achieves a small generalization error, even when the network has enough capacity to fit arbitrary labels
- SGD has a strong inductive bias on structured data: finds good generalization guarantees instead of finding bad global optima that can fit arbitrary labels

# Questions need to be addressed

1. Why can SGD optimize the training loss? Or even find a critical point?
2. Why can the trained network generalize?

# Observations

- When the network is overparameterized, it becomes more pseudo smooth, which makes easier for SGD to minimize the training loss.
- **Observation:** The more we overparameterize the network, the less likely the activation pattern for one neuron and one data point will change in a fixed number of iterations.
    - allows us to couple the gradient of the true neural network with a “pseudo gradient” where the activation pattern for each data point and each neuron is fixed

# Pseudo Gradient

- pseudo gradient for fixed r, i whether the r-th hidden node is activated on the i-th data point $x_i$ will always be the same for different t
- But for fixed t, for different r or i, the sign can be different.
- ***To be proved:***
  - Unless the generalization error is small, the pseudo gradient will always be large
  - As number m of hidden neurons increases, with a properly decreasing learning rate, the total number of iterations it takes to minimize the loss is roughly not changed
  - Number of iterations that we can couple the true gradient with the pseudo one increases. Hence, there is a polynomially large m so that we can couple these two gradients until the network reaches a small generalization error.

# Simplified Case: No Variance

**Assumption:** Each $D_{a,b}$ is a single data point ($x_{a,b}$, a), and also we are doing full batch gradient descent as opposite to the minibatch SGD.

***Loss Notation:*** $L(w) = \sum_{a\in[k],b\in[l]} p_{a,b} L(w, x_{a,b}, a)$

***Gradient:*** $$\frac{\partial L(w)}{\partial w_r} = \sum_{a\in[k],b\in[l]} p_{a,b} \left( \sum_{i\neq a} a_{i,r} o_i(x_{a,b}, w) - \sum_{i\neq a} a_{a,r} o_i(x_{a,b}, w) \right) 1_{\langle w_r, x_{a,b}\rangle \geq 0} x_{a,b}.$$

***Pseudo Gradient:*** $$\frac{\tilde{\partial} L(w)}{\partial w_r} = \sum_{a\in[k],b\in[l]} p_{a,b} \left( \sum_{i\neq a} a_{i,r} o_i(x_{a,b}, w) - \sum_{i\neq a} a_{a,r} o_i(x_{a,b}, w) \right) 1_{\langle w_r^{(0)}, x_{a,b}\rangle \geq 0} x_{a,b}$$

# Contd...

- For pseudo gradient, the activation pattern is set to be that in the initialization
- the pseudo gradient is similar to the gradient for a pseudo network g defined as: $g_i(x, \breve{w}) := \sum_{r=1}^{m} a_{i,r} \langle w_r, x \rangle \breve{1}_{\left\langle w_r^{(0)}, x \right\rangle \geq 0}$
- Coupling the gradients is similar to coupling the networks f and g

# Lemma

1. At each iteration, the total number of hidden units whose gradient can be coupled with the pseudo one is quite large

**Lemma 5.1** (Coupling). *W.h.p. over the random initialization, for every* $\tau > 0$, *for every* $t = \tilde{O}\left(\frac{\tau}{\eta}\right)$, *we have that for at least* $1 - \frac{e\tau kl}{\sigma}$ *fraction of* $r \in [m]$: $\frac{\partial L(w^{(t)})}{\partial w_r} = \frac{\tilde{\partial} L(w^{(t)})}{\partial w_r}$.

2. Pseudo gradient is large unless the error is small

**Lemma 5.2.** *For* $m = \tilde{\Omega}\left(\frac{k^3 l^2}{\delta}\right)$, *for every* $\{p_{a,b} v_{i,a,b}\}_{i,a\in[k],b\in[l]} \in [-v, v]$ *(that depends on* $w_r^{(0)}, a_{i,r}$, *etc.) with* $\max\{p_{a,b} v_{i,a,b}\}_{i,a\in[k],b\in[l]} = v$, *there exists at least* $\Omega(\frac{\delta}{kl})$ *fraction of* $r \in [m]$ *such that* $\left\| \frac{\tilde{\partial} L(w)}{\partial w_r} \right\|_2 = \tilde{\Omega}\left(\frac{v\delta}{kl}\right)$.

This paper illustrates how to use these two lemmas to show the convergence for a small enough learning rate

# Classification error

We define

$$v_{s,a,b}(w) = \begin{cases} \frac{\sum_{i \neq a} e^{f_i(x_{a,b},w)}}{\sum_{i=1}^{k} e^{f_i(x_{a,b},w)}} & \text{if } s = a; \\ -\frac{e^{f_s(x_{a,b},w)}}{\sum_{i=1}^{k} e^{f_i(x_{a,b},w)}} & \text{otherwise.} \end{cases}$$

$V_{a,a,b}$ indicates the classification error

**By definition, $v_{i,a,b}$'s satisfy:**

1. $\forall a \in [k], b \in [l] : v_{a,a,b} \in [0, 1]$
2. $\sum_{i=1}^{k} v_{i,a,b} = 0.$

# Proof of Coupling

**Lemma A.1** (Coupling, Lemma 5.1 restated)**.** *W.h.p. over the random initialization, for every* $\tau > 0$*, for every* $t = \tilde{O}\left(\frac{\tau}{\eta}\right)$*, we have that for at least* $1 - \frac{e\tau kl}{\sigma}$ *fraction of* $r \in [m]$*:*

$$\frac{\partial L(w^{(t)})}{\partial w_r} = \frac{\tilde{\partial} L(w^{(t)})}{\partial w_r}.$$

*Proof.* W.h.p. we know that every $|a_{i,r}| \leq L = \tilde{O}(1)$. Thus, for every $r \in [m]$ and every $t \geq 0$, we have

$$\left\| \frac{\partial L(w^{(t)})}{\partial w_r} \right\|_2 \leq L$$

which implies that $\left\| w_r^{(t)} - w_r^{(0)} \right\|_2 \leq L\eta t$.

# Error, Gradient Proof

Pseudo gradient can be written as:

$$\frac{\tilde{\partial} L(w)}{\partial w_r} = \sum_{i \in [k]} a_{i,r} P_{i,r}$$

$$P_{i,r} = \sum_{a \in [k], b \in [l]} p_{a,b} v_{i,a,b} \mathbf{1}_{\langle w_r^{(0)}, x_{a,b} \rangle \geq 0} x_{a,b}.$$

if $p_{a,b} v_{i,a,b}$ is large, a good fraction of $r \in [m]$ will have large pseudo gradient

# Convergence Proof

**Lemma A.4** (Convergence). *Let us denote* $\max\{p_{a,b}v_{i,a,b}^{(t)}\} = v^{(t)}$. *Then for a sufficiently small* $\eta$, *we have that for every* $T = \tilde{\Theta}\left(\frac{\sigma\delta}{kl\eta}\right)$,

$$\frac{1}{T}\sum_{t=1}^{T}\left(v^{(t)}\right)^2 = \tilde{O}\left(\frac{k^5l^5}{\delta^4\sigma m}\right).$$

By our choice of $\sigma = \tilde{O}\left(\frac{1}{m^{1/2}}\right)$, we know that

$$\frac{1}{T}\sum_{t=1}^{T}\left(v^{(t)}\right)^2 = \tilde{O}\left(\frac{k^5l^5}{\delta^4 m^{1/2}}\right)$$

This shows that eventually $v^{(t)}$ will be small that leads to small classification error

# Coupling, Low Complexity

**Coupling:** how well components of a system depend on each other

- **Benefits of Tight Coupling:**
    - Components in the system are highly dependent on each other
    - Can lead to improve performance, as the system can be optimized as a whole instead of optimizing individual components separately

**Low Complexity:** refers to the models that have relatively small number of parameters and layers

# Weight Compression

**Weight Compression:**

- Technique used in Neural network to reduce the size of the model
- It decreases the computational cost
- Weights of the neurons which doesn’t have much significance is reduced

# Insights from Analysis

- **Generalization:**
  - Our analysis partially explained how SGD on structured data leads to low complexity
  - SGD can reduce the error and reach a good solution
  - Closeness to the initialization means the weights can be easily compressed
  - Showed that there can be a solution not far from the initialization with high probability
  - When data is well clustered around few patterns, the accumulated updates (difference between the learned weights and the initialization) should be approximately low rank

# Contd

- **Implicit regularization v.s. structure of the data:**
  - Existing work has analyzed the implicit regularization of SGD on linearly separable data
  - Our analysis shows that when the network size is fixed, learning over poorly structured data(large k and l) needs more iterations  and thus the solution can deviate more from the initialization and has higher complexity
  - An interesting case is if can fit the training data by viewing each point as a component, results show that it still learns a network with a small generalization error
- **Effect of random initialization:**
  - Analysis shows how proper random initializations helps the optimization and generalization
  - With high probability for weights close to the initialization, SGD makes progress when the loss is large and eventually learns a good solution
  - Our initialization has scale related to the number of hidden units, which is useful when network size is varying

# Experiments

Experiments are performed on synthetic data and MNIST datasets to verify:

1. The activation patterns of the hidden units couple with those at initialization
2. The distance from the learned solution is relatively small compared to size of initialization
3. The accumulated updates have approximately low rank

**Setup:** Synthetic data is of 1000 dimension and contains 10 classes each with 2 components. Each component is of equal probability and is a gaussian distribution.

***On Synthetic Data:*** SGD is run for T=400 steps with batch size B=16 and learning rate $\eta = 10/m$

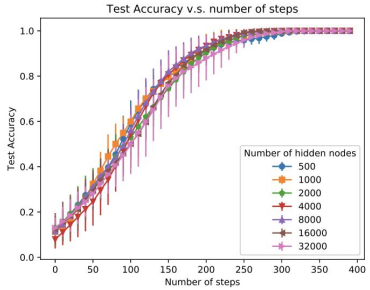
***On MNIST:*** SGD is run for T=2x10^4 steps with batch size B=64 and learning rate $\eta = 400/m$

*m: number of hidden units*

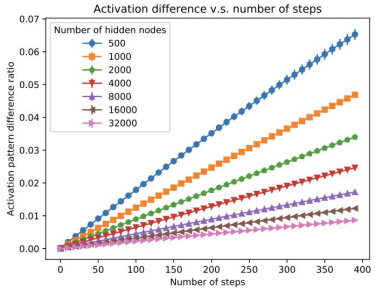
# Evaluation metrics

1. Test accuracy
2. **Coupling:** fraction of hidden units whose activation pattern changed compared to the time at initialization
3. **Distance:** relative ratio $||w^{(t)} - w^{(0)}||_F / ||w^{(0)}||_F$
4. **Rank of accumulated updates:** plot of the singular values of $w^{(T)} - w^{(0)}$, where T is the final step

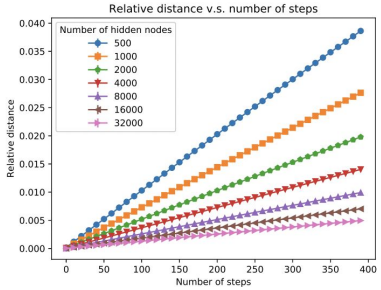
# Results on Synthetic Data



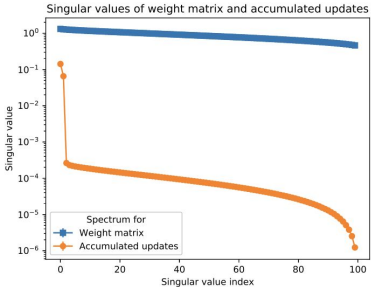

(a) Test accuracy




(b) Coupling




(c) Distance from the initialization

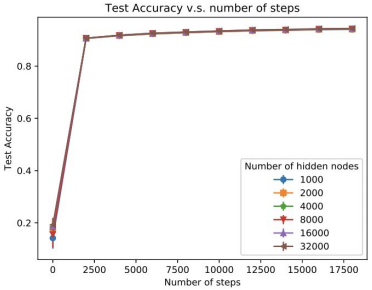



(d) Rank of accumulated updates ($y$-axis in log-scale)

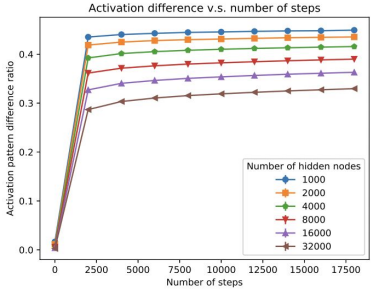
Figure 1: Results on the synthetic data.

1. Accuracy quickly converges to 100% which proving overparameterization helps optimization and generalization
2. Strong coupling as the activation pattern difference ratio is less than 0.1
3. Relative distance is less than 0.1 which shows final solution is close to initialization
4. The top 20 singular values of the accumulated updates are much larger than the rest

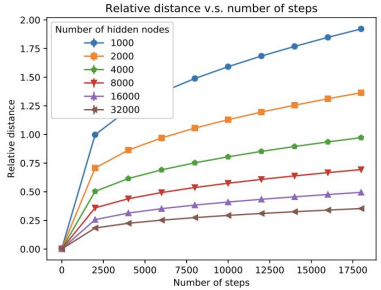
# Results on MNIST dataset



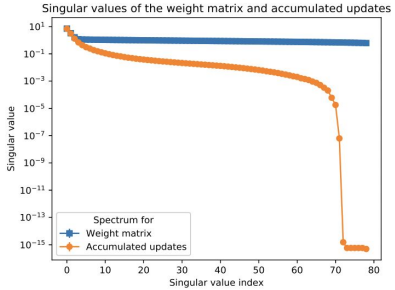

(a) Test accuracy




(b) Coupling




(c) Distance from the initialization




(d) Rank of accumulated updates ($y$-axis in log-scale)

Figure 2: Results on the MNIST data.

Results on MNIST are similar to that of synthetic data.

We can observe that the trend becomes more evident with more overparameterization.

# Proofs for General Case

**Coupling:**

**Lemma B.1** (Coupling). *For every unit vector $x \in \mathbb{R}^d$, w.h.p. over the random initialization, for every $\tau > 0$, for every $t = \tilde{O}\left(\frac{\tau}{\eta}\right)$ we have that for at least $1 - \frac{10\tau}{\sigma}$ fraction of $r \in [m]$:*

$$\frac{\partial L(w^{(t)}, x, y)}{\partial w_r} = \frac{\tilde{\partial} L(w^{(t)}, x, y)}{\partial w_r} (\forall y \in [k]), \quad \text{and} \quad |\langle w_r^{(t)}, x \rangle| \geq \tau.$$

Proof is similar to simplified case

# Error, Gradient Proof

**Lemma B.3.** *For every $v > 0$, for $m = \tilde{\Omega}\left(\left(\frac{kl}{v\delta}\right)^4\right)$, for every possible $\{p_{a,b}v_{i,a,b}\}$ (that depend on $a_{i,r}, w_r^{(0)}$, etc.) such that $\max\{\mathbb{E}[p_{a,b}v_{i,a,b}]\}_{i,a\in[k],b\in[l]} = v$, there exists at least $\Omega\left(\frac{\delta}{kl}\right)$ fraction of $r \in [m]$ such that*

$$\left\| \frac{\tilde{\partial} L(w)}{\partial w_r} \right\|_2 = \tilde{\Omega}\left(\frac{v\delta}{kl}\right).$$

This lemma implies that if the classification error is large, then many $w_r$'s have a large pseudo gradient.

# Convergence

**Lemma B.4** (Convergence). *Denote* $\max\{\mathbb{E}[p_{a,b}v_{i,a,b}(x_{a,b}, w^{(t)})]\}_{i,a\in[k],b\in[\ell]} = v^{(t)} = v$, *and let* $\gamma = \Omega\left(\frac{\delta}{kl}\right)$. *Then for a sufficiently small* $\eta = \tilde{O}\left(\frac{\gamma}{m}\left(\frac{v\delta}{kl}\right)^2\right)$, *if we run SGD with a batch size at least* $B_t = \tilde{\Omega}\left(\left(\frac{kl}{v\delta}\right)^4 \frac{1}{\gamma^2}\right)$ *and* $t = \tilde{O}\left(\left(\frac{v\delta}{kl}\right)^2 \frac{\sigma\gamma}{\eta}\right)$, *then w.h.p.,*

$$L(w^{(t)}) - L(w^{(t+1)}) = \eta\gamma m\tilde{\Omega}\left(\left(\frac{v\delta}{kl}\right)^2\right).$$

*Proof of Lemma B.4.* We know that for at least $\gamma$ fraction of $r \in [m]$ such that

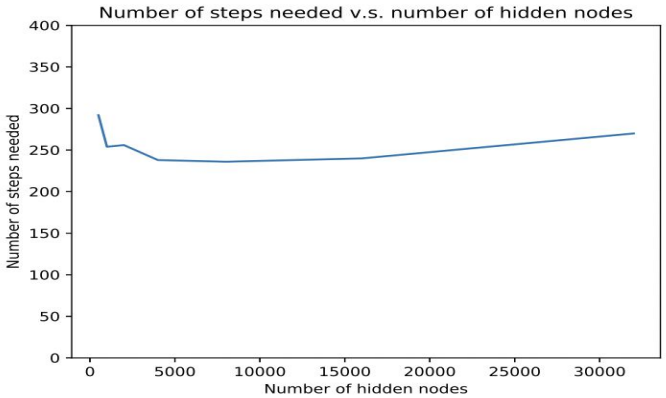
$$\left\|\frac{\tilde{\partial}L(w^{(t)})}{\partial w_r}\right\|_2 = \tilde{\Omega}\left(\frac{v\delta}{kl}\right).$$

At the end, We need $\frac{\sigma}{\eta}\frac{\delta^3\varepsilon^6}{k^5\ell^5} = \tilde{\Omega}\left(\frac{1}{\eta m}\frac{k^5\ell^5}{\delta^3\varepsilon^6}\right)$

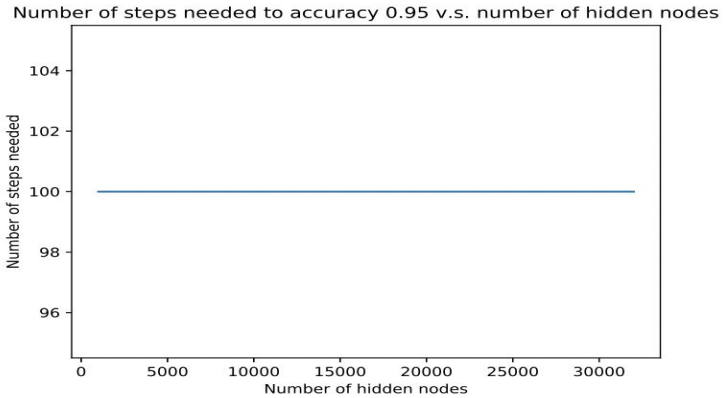
To keep coupling before convergence

# Additional Experimental Results

We discussed that for a learning rate decreasing with the number of hidden nodes m, the number of iterations to get the accuracy roughly remain the same




(a) On synthetic data

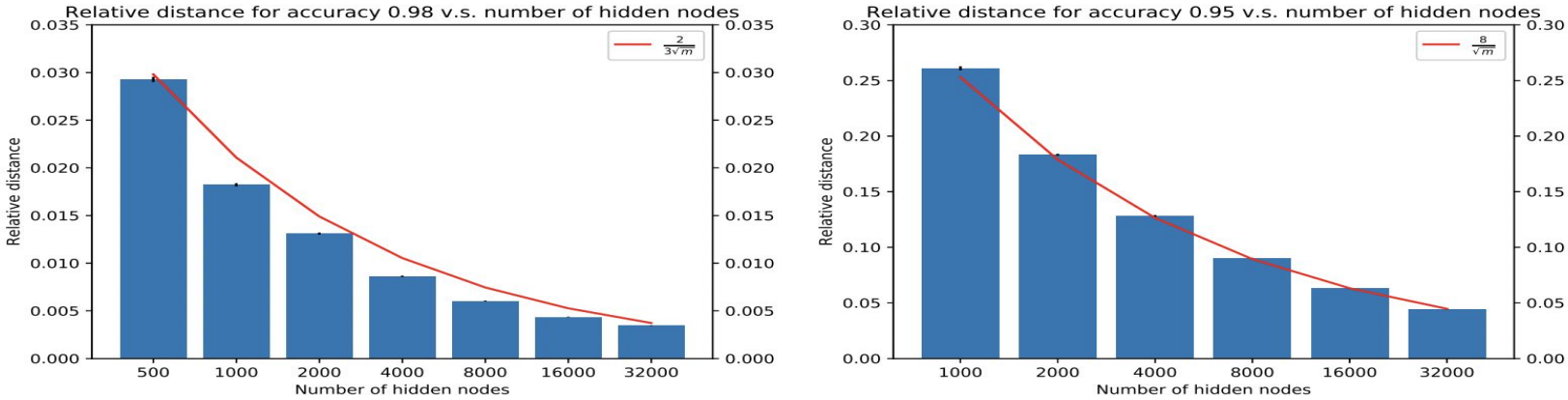



(b) on MNIST

Figure 4: Number of steps to achieve 98% on the synthetic data and 95% test accuracy on MNIST for different values of number of hidden nodes. They are roughly the same for different number of hidden nodes.

# Contd

It is also observed that the relative distances scale roughly as O(1/ √ m)

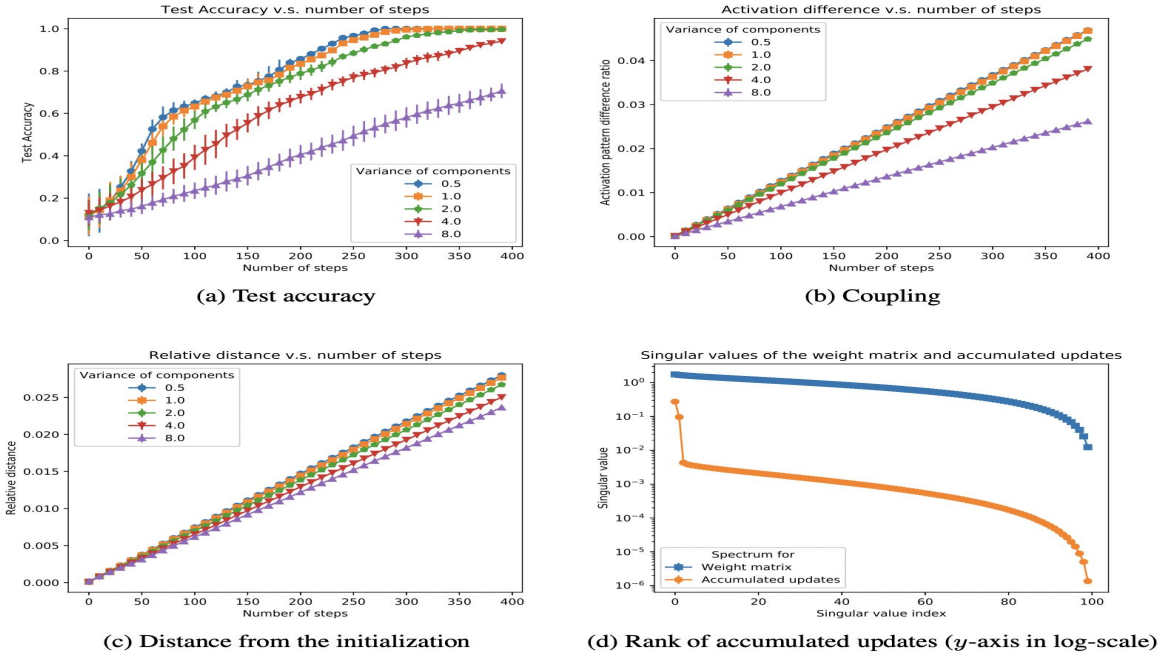



(a) On synthetic data

(b) on MNIST

Figure 5: Relative distances when achieving 98% on the synthetic data and 95% test accuracy on MNIST for different values of number of hidden nodes. They closely match $2/3\sqrt{m}$ on the synthetic data and $8/\sqrt{m}$ on MNIST (the red lines), where $m$ is the number of hidden nodes.

# Synthetic Data with larger Variances



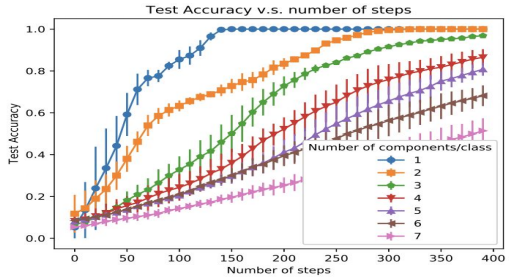

(a) Test accuracy (b) Coupling

(c) Distance from the initialization (d) Rank of accumulated updates ($y$-axis in log-scale)

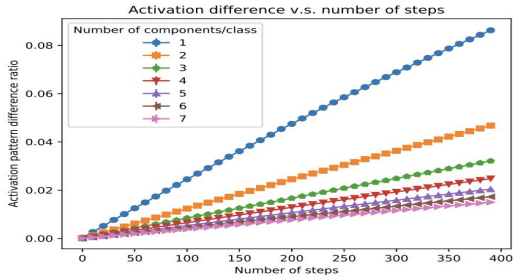
Figure 6: Results for synthetic data with different variances.

- Test accuracy decreases with increase in variance
- No change in trends for activation patterns, distance, and the rank of the weight matrix (maybe because signal in updates remains small with increasing variances)

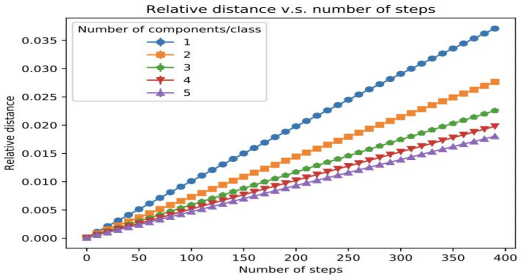
# Synthetic Data with Large number of components in each class



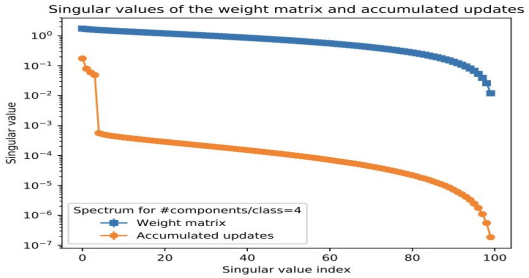

(a) Test accuracy




(b) Coupling




(c) Distance from the initialization




(d) Rank of accumulated updates for 4 components in each class ($y$-axis in log-scale)

Figure 7: Results for synthetic data with larger number of components in each class.

- Test accuracy decreases with increase in number of components
- Larger l leads to more significant coupling and small relative distances (maybe because learning makes less progress due to more complicated structured data)

# Conclusion

- We studied the problem of learning with two layer overparameterized neural network via SGD
- Our work is far from being conclusive
- We made a step towards theoretical understanding of SGD for training neural networks

# Thank You!