Търсене и извличане на информация. Приложение на дълбоко машинно обучение

Стоян Михов

Лекция 11: Не-марковски езиков модел. Рекурентни невронни мрежи. Архитектури на рекурентни невронни мрежи с портали

План на лекцията

1. **Формалности за курса (5 мин)**
2. Не-марковски рекурентен езиков модел (10 мин)
3. Пропагиране напред при рекурентна невронна мрежа (5 мин)
4. Пропагиране назад при рекурентна невронна мрежа (5 мин)
5. Особености при обучение на рекурентна невронна мрежа (30 мин)
6. Проблем и решение при експлодиращ градиент (10 мин)
7. Проблем и архитектури за решаване на проблема при изчезващ градиент (25 мин)

Формалности

- В Moodle ще намерите оценките и решенията на Домашно задание №1
- В Moodle ще бъде публикувано Домашно задание №2 към края на следващата седмица
- Лекция 11 се базира на глава 14 от втория учебник.

План на лекцията

1. Формалности за курса (5 мин)
2. **Не-марковски рекурентен езиков модел (10 мин)**
3. Пропагиране напред при рекурентна невронна мрежа (5 мин)
4. Пропагиране назад при рекурентна невронна мрежа (5 мин)
5. Особености при обучение на рекурентна невронна мрежа (30 мин)
6. Проблем и решение при експлодиращ градиент (10 мин)
7. Проблем и архитектури за решаване на проблема при изчезващ градиент (25 мин)

Марковски k-грамен невронен езиков модел

Миналата лекция разгледахме модела на Бенджио и съавтори:

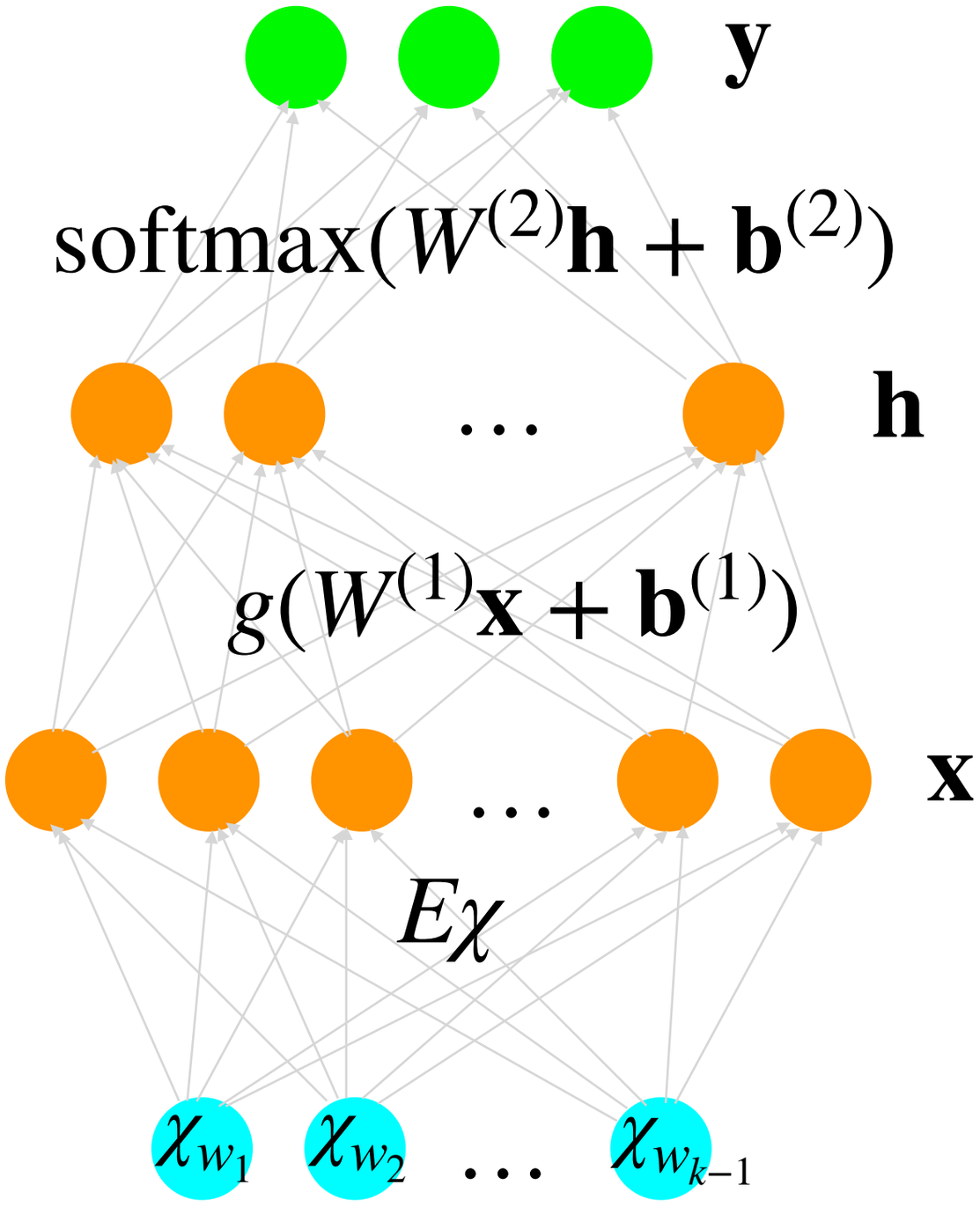
$$\mathbf{y} = \text{softmax}(W^{(2)}\mathbf{h} + \mathbf{b}^{(2)})$$

$$\mathbf{h} = g(W^{(1)}\mathbf{x} + \mathbf{b}^{(1)})$$

$$\mathbf{x} = \begin{bmatrix} E\chi_{w_1} \\ \vdots \\ E\chi_{w_{k-1}} \end{bmatrix}$$

Езиков модел:

$$\Pr[w \mid w_1 w_2 \ldots w_{k-1}] = \mathbf{y}_w$$



Проблем: Зависимости на голямо разстояние

- Пример:

  Книгата, която всички търсеха и толкова много харесваха, беше най-после върната от учителя по литература в [redacted]

  Котката, която всички търсеха и толкова много харесваха, беше най-после върната от учителя по литература в [redacted]

Проблем: Зависимости на голямо разстояние

- Пример:

  Книгата, която всички търсеха и толкова много харесваха, беше най-после върната от учителя по литература в библиотеката.

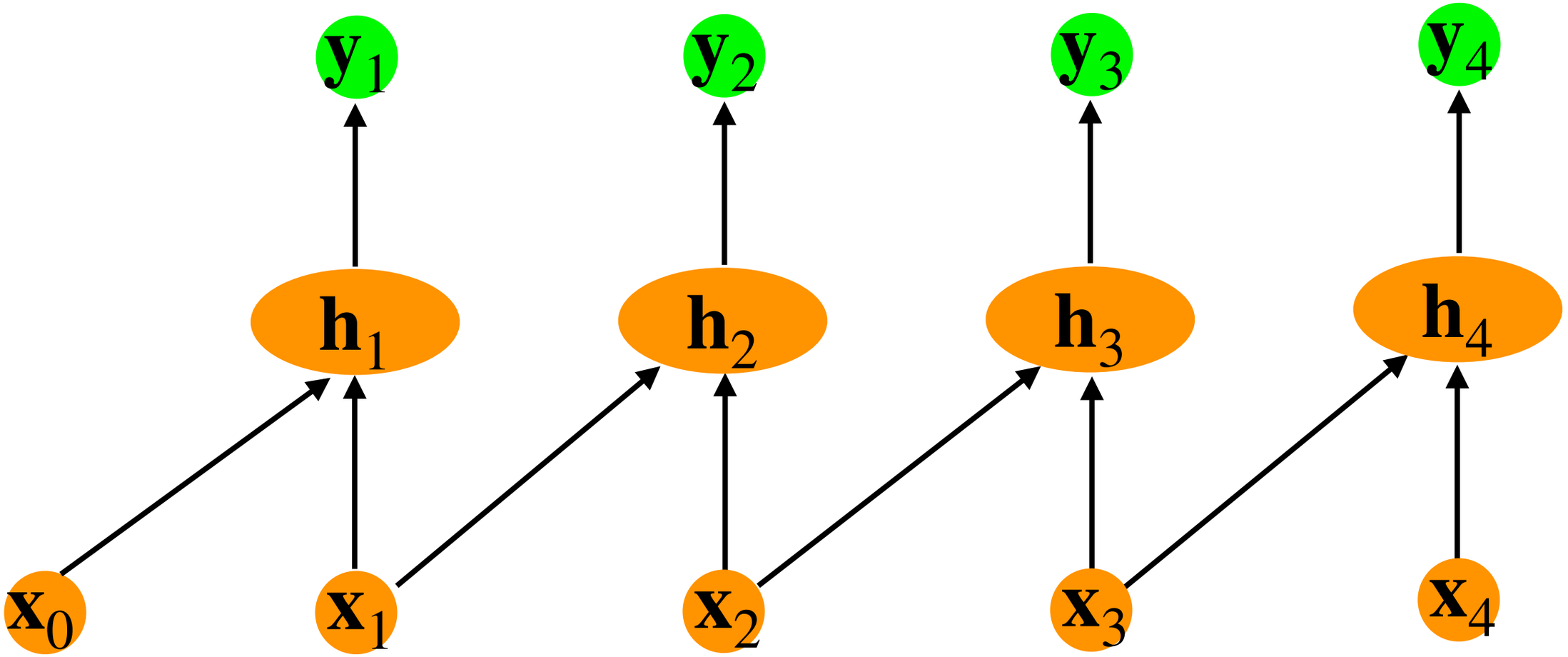
  Котката, която всички търсеха и толкова много харесваха, беше най-после върната от учителя по литература в приюта.

- Разстоянието между *книгата* и *библиотеката или котката и приюта* е 15 думи.

- Използването на k-грамен езиков модел при $k < 15$ не може да даде за този случай задоволителен резултат.

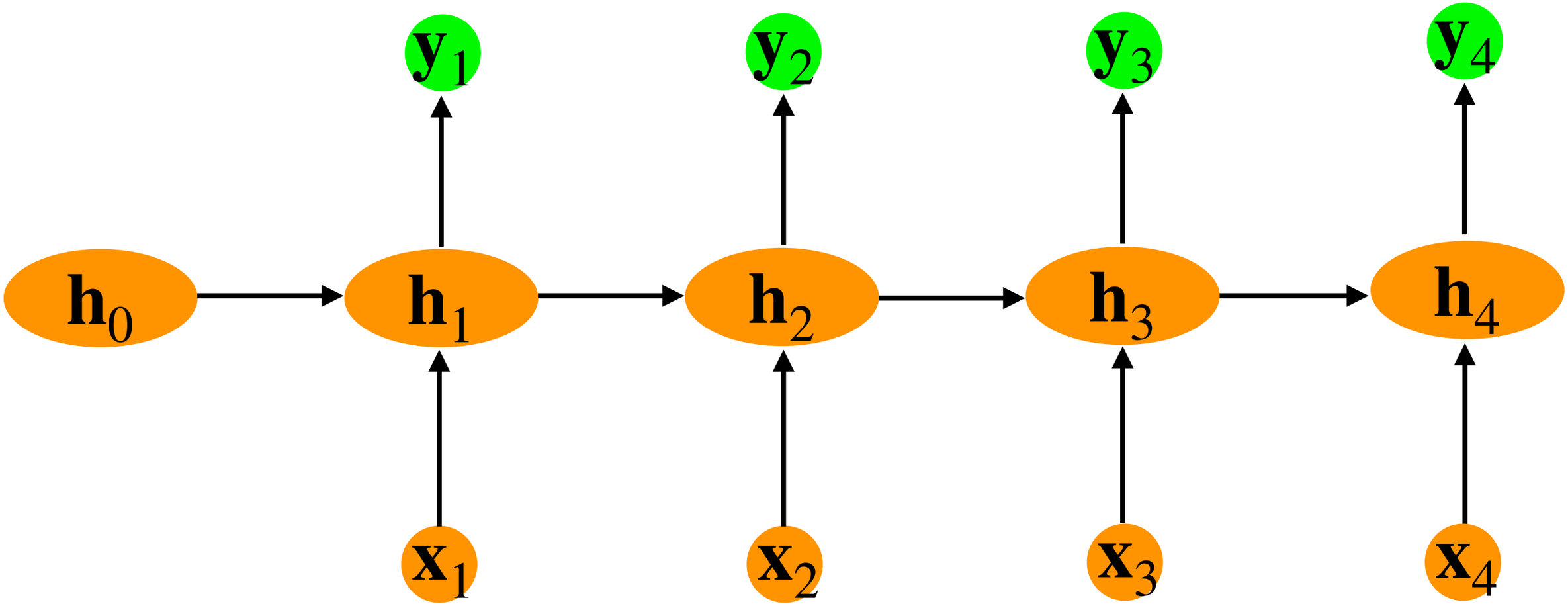
Рекурентни невронни мрежи

- Нека разгледаме триграмен невронен езиков модел.
- Векторите $\mathbf{h}$ представляват влагане на контекста, от което се получава вероятностно разпределение $\mathbf{y}$.
- Може ли да получим новия контекст от предишния, като го допълним със следващата дума?



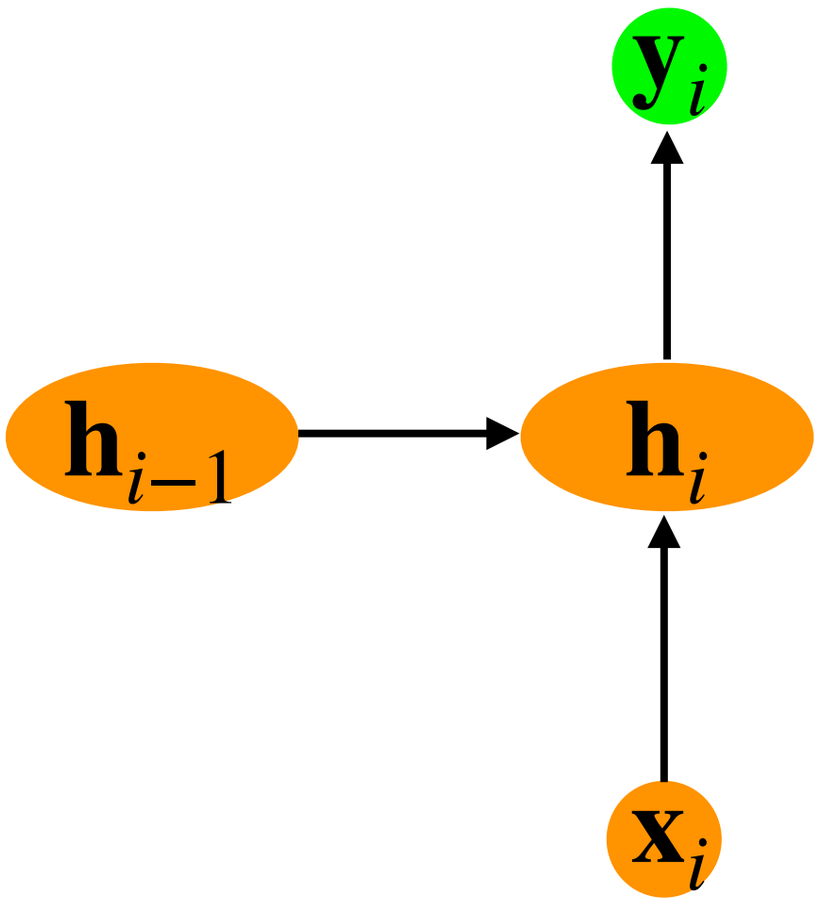
Рекурентни невронни мрежи

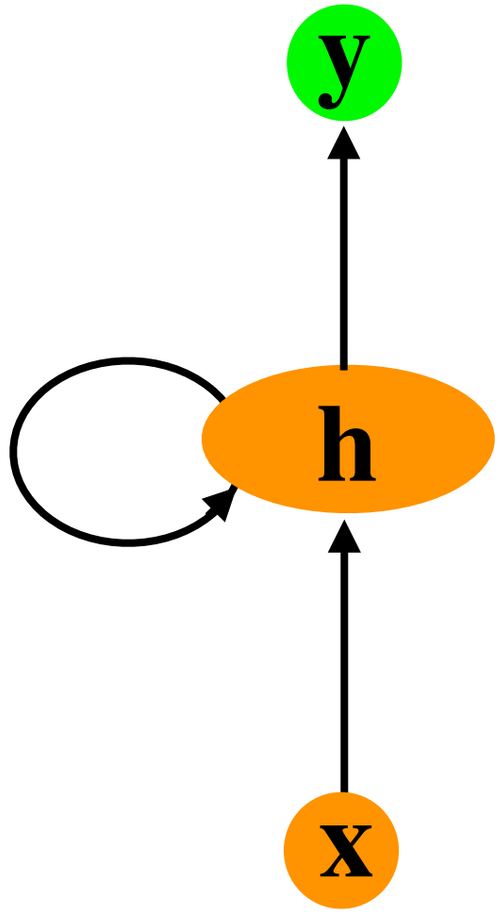
- Искаме да натрупваме към контекста до момента новата дума.



Рекурентни невронни мрежи

- Нека да се абстрахираме от поредния номер





Прости рекурентни невронни мрежи: Elman RNN

$$\mathbf{y}_i = \text{softmax}(U\mathbf{h}_i)$$

$$\mathbf{h}_i = g(W\mathbf{h}_{i-1} + V\mathbf{x}_i)$$

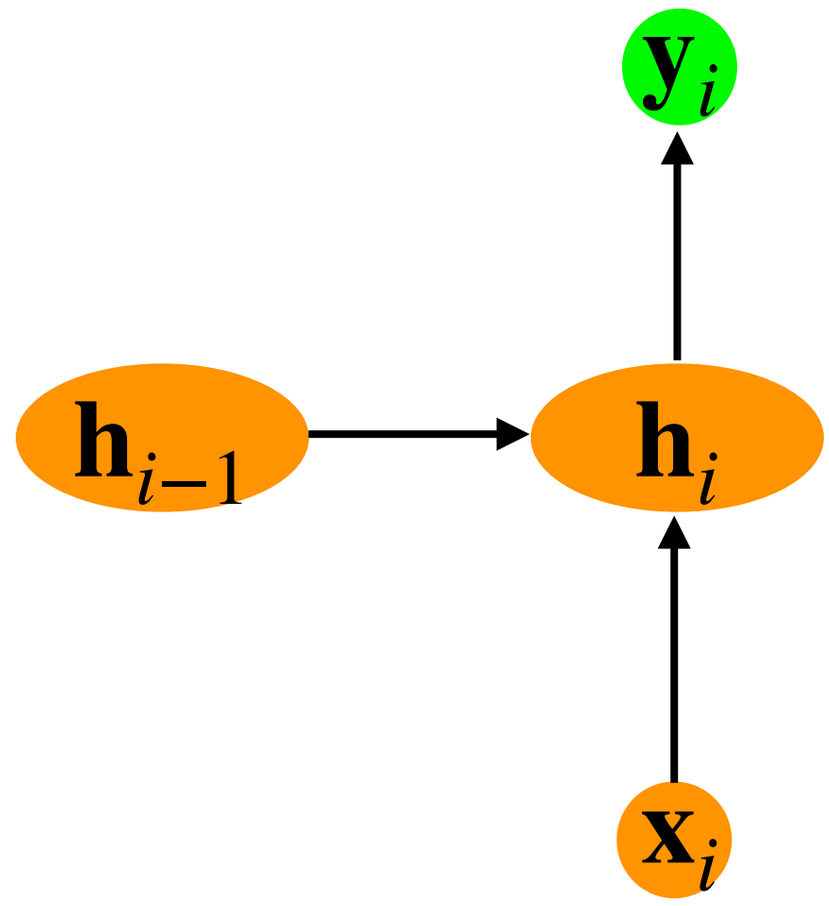
$$\mathbf{x}_i = E\chi_{w_i}$$

$$\chi_{w_i} \in \mathbb{R}^{|L|}, E \in \mathbb{R}^{M \times |L|},$$

$$\mathbf{x}_i \in \mathbb{R}^M, V \in \mathbb{R}^{N \times M},$$

$$\mathbf{h}_i, \mathbf{h}_{i-1} \in \mathbb{R}^N, W \in \mathbb{R}^{N \times N},$$

$$U \in \mathbb{R}^{|L| \times N}, \mathbf{y}_i \in \mathbb{R}^{|L|}$$



Езиков модел с рекурентна невронна мрежа

При входен текст $w_1 w_2 \ldots w_n$ с произволна дължина n, моделираме вероятностното разпределение за следващата дума като:

$$\Pr_{E,V,W,U}[w \mid w_1 w_2 \ldots w_n] = (\mathbf{y}_n)_w, \text{ където}$$

- $\mathbf{y}_i = \text{softmax}(U\mathbf{h}_i)$
- $\mathbf{h}_i = g(W\mathbf{h}_{i-1} + VE\chi_{w_i})$, за $i = 1,2,\ldots,n$, $\mathbf{h}_0$ - фиксирано

Стойността на $\Pr_{E,V,W,U}[w \mid w_1 w_2 \ldots w_n] = (\mathbf{y}_n)_w$ ще зависи от **всички** предходни думи $w_1 w_2 \ldots w_n$

Езиков модеп с рекурентна невронна мрежа избягва Марковското ограничение

План на лекцията

1. Формалности за курса (5 мин)
2. Не-марковски рекурентен езиков модел (10 мин)
3. **Пропагиране напред при рекурентна невронна мрежа (5 мин)**
4. Пропагиране назад при рекурентна невронна мрежа (5 мин)
5. Особености при обучение на рекурентна невронна мрежа (30 мин)
6. Проблем и решение при експлодиращ градиент (10 мин)
7. Проблем и архитектури за решаване на проблема при изчезващ градиент (25 мин)

Пропагиране напред при рекурентна невронна мрежа

Параметри: $E \in \mathbb{R}^{M \times |L|}, V \in \mathbb{R}^{N \times M}, W \in \mathbb{R}^{N \times N}, U \in \mathbb{R}^{|L| \times N}$

Вход: $\mathbf{h}_0 \in \mathbb{R}^N, \chi_{w_1}, \chi_{w_2}, \chi_{w_3}, \ldots \in \mathbb{R}^{|L|}$

Пропагиране напред при рекурентна невронна мрежа

Параметри: $E \in \mathbb{R}^{M \times |L|}, V \in \mathbb{R}^{N \times M}, W \in \mathbb{R}^{N \times N}, U \in \mathbb{R}^{|L| \times N}$

Вход: $\mathbf{h}_0 \in \mathbb{R}^N, \chi_{w_1}, \chi_{w_2}, \chi_{w_3}, \ldots \in \mathbb{R}^{|L|}$

$$\mathbf{x}_1 = E\chi_{w_1}, \quad \mathbf{h}_1 = g(W\mathbf{h}_0 + V\mathbf{x}_1), \quad \mathbf{y}_1 = \text{softmax}(U\mathbf{h}_1)$$

Пропагиране напред при рекурентна невронна мрежа

Параметри: $E \in \mathbb{R}^{M \times |L|}, V \in \mathbb{R}^{N \times M}, W \in \mathbb{R}^{N \times N}, U \in \mathbb{R}^{|L| \times N}$

Вход: $\mathbf{h}_0 \in \mathbb{R}^N, \chi_{w_1}, \chi_{w_2}, \chi_{w_3}, \ldots \in \mathbb{R}^{|L|}$

$$\mathbf{x}_1 = E\chi_{w_1}, \quad \mathbf{h}_1 = g(W\mathbf{h}_0 + V\mathbf{x}_1), \quad \mathbf{y}_1 = \text{softmax}(U\mathbf{h}_1)$$

$$\mathbf{x}_2 = E\chi_{w_2}, \quad \mathbf{h}_2 = g(W\mathbf{h}_1 + V\mathbf{x}_2), \quad \mathbf{y}_2 = \text{softmax}(U\mathbf{h}_2)$$

Пропагиране напред при рекурентна невронна мрежа

Параметри: $E \in \mathbb{R}^{M \times |L|}, V \in \mathbb{R}^{N \times M}, W \in \mathbb{R}^{N \times N}, U \in \mathbb{R}^{|L| \times N}$

Вход: $\mathbf{h}_0 \in \mathbb{R}^N, \chi_{w_1}, \chi_{w_2}, \chi_{w_3}, \ldots \in \mathbb{R}^{|L|}$

$$\mathbf{x}_1 = E\chi_{w_1}, \quad \mathbf{h}_1 = g(W\mathbf{h}_0 + V\mathbf{x}_1), \quad \mathbf{y}_1 = \text{softmax}(U\mathbf{h}_1)$$

$$\mathbf{x}_2 = E\chi_{w_2}, \quad \mathbf{h}_2 = g(W\mathbf{h}_1 + V\mathbf{x}_2), \quad \mathbf{y}_2 = \text{softmax}(U\mathbf{h}_2)$$

$$\mathbf{x}_3 = E\chi_{w_3}, \quad \mathbf{h}_3 = g(W\mathbf{h}_2 + V\mathbf{x}_3), \quad \mathbf{y}_3 = \text{softmax}(U\mathbf{h}_3)$$

$$\vdots$$

Пропагиране напред при рекурентна невронна мрежа

Параметри: $E \in \mathbb{R}^{M \times |L|}, V \in \mathbb{R}^{N \times M}, W \in \mathbb{R}^{N \times N}, U \in \mathbb{R}^{|L| \times N}$

Вход: $\mathbf{h}_0 \in \mathbb{R}^N, \chi_{w_1}, \chi_{w_2}, \chi_{w_3}, \ldots \in \mathbb{R}^{|L|}$

$$\mathbf{x}_1 = E\chi_{w_1}, \quad \mathbf{h}_1 = g(W\mathbf{h}_0 + V\mathbf{x}_1), \quad \mathbf{y}_1 = \text{softmax}(U\mathbf{h}_1)$$

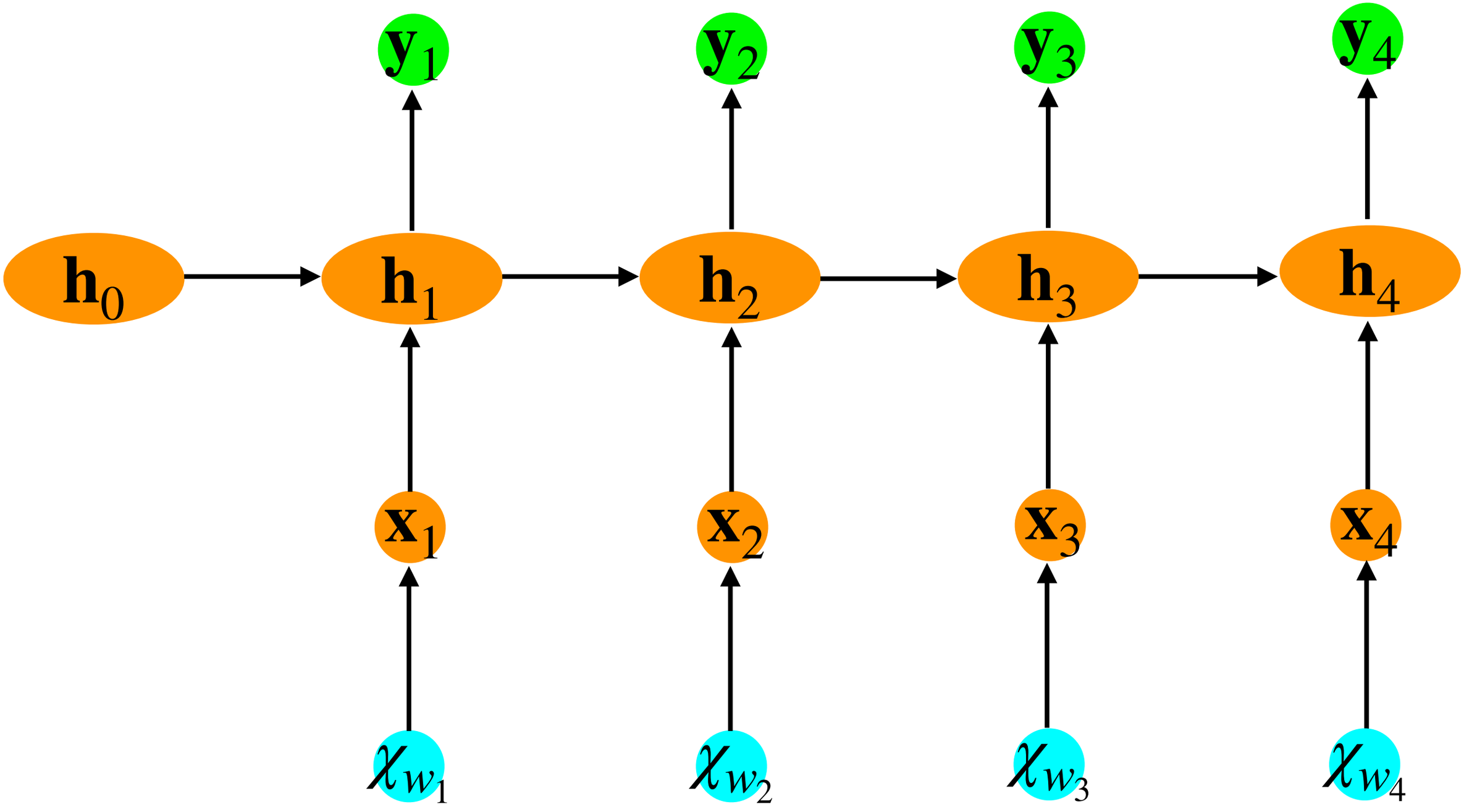
$$\mathbf{x}_2 = E\chi_{w_2}, \quad \mathbf{h}_2 = g(W\mathbf{h}_1 + V\mathbf{x}_2), \quad \mathbf{y}_2 = \text{softmax}(U\mathbf{h}_2)$$

$$\mathbf{x}_3 = E\chi_{w_3}, \quad \mathbf{h}_3 = g(W\mathbf{h}_2 + V\mathbf{x}_3), \quad \mathbf{y}_3 = \text{softmax}(U\mathbf{h}_3)$$

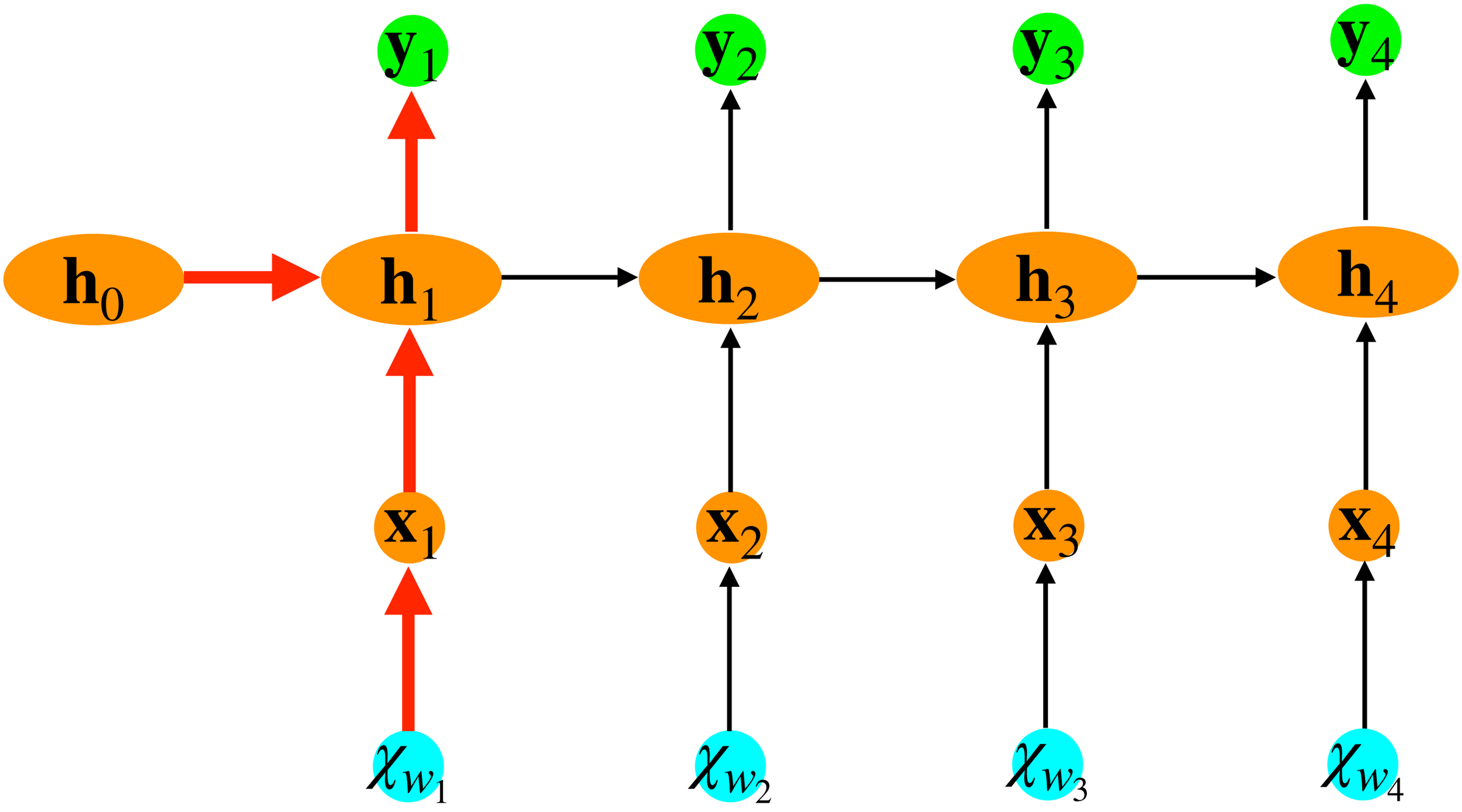
$\vdots$

Изход: $\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \ldots \in \mathbb{R}^{|L|}$

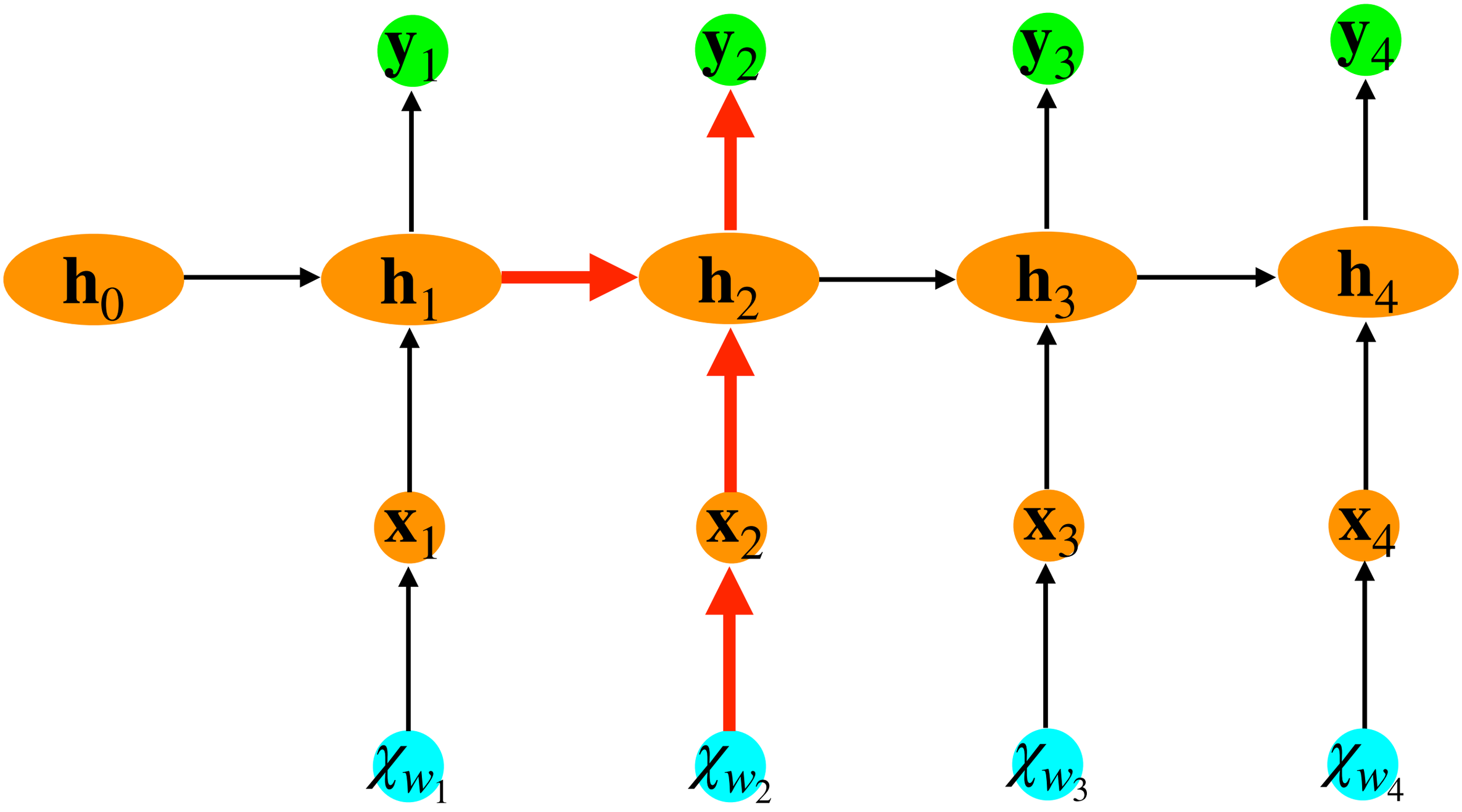
Пропагиране напред при рекурентна невронна мрежа



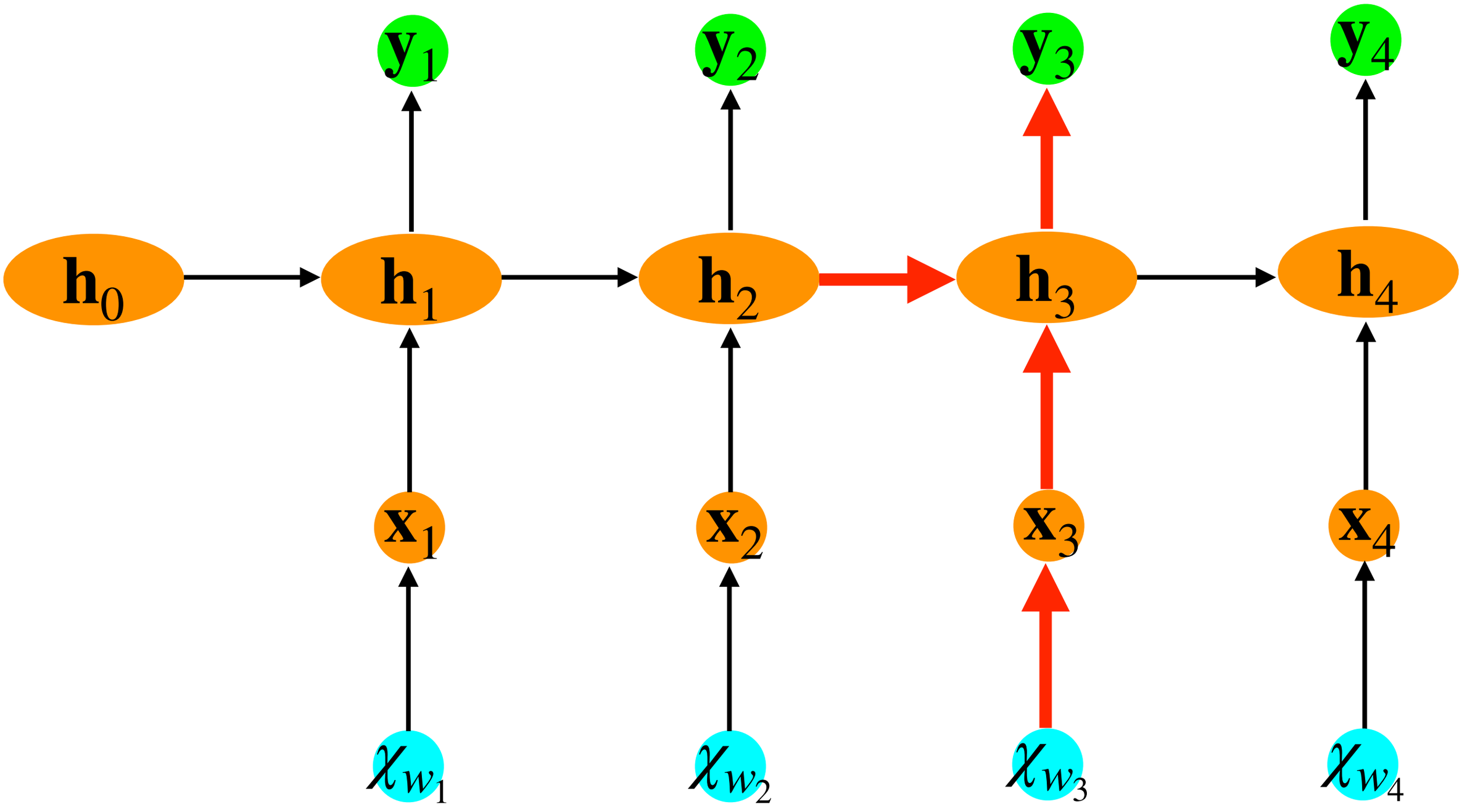
Пропагиране напред при рекурентна невронна мрежа



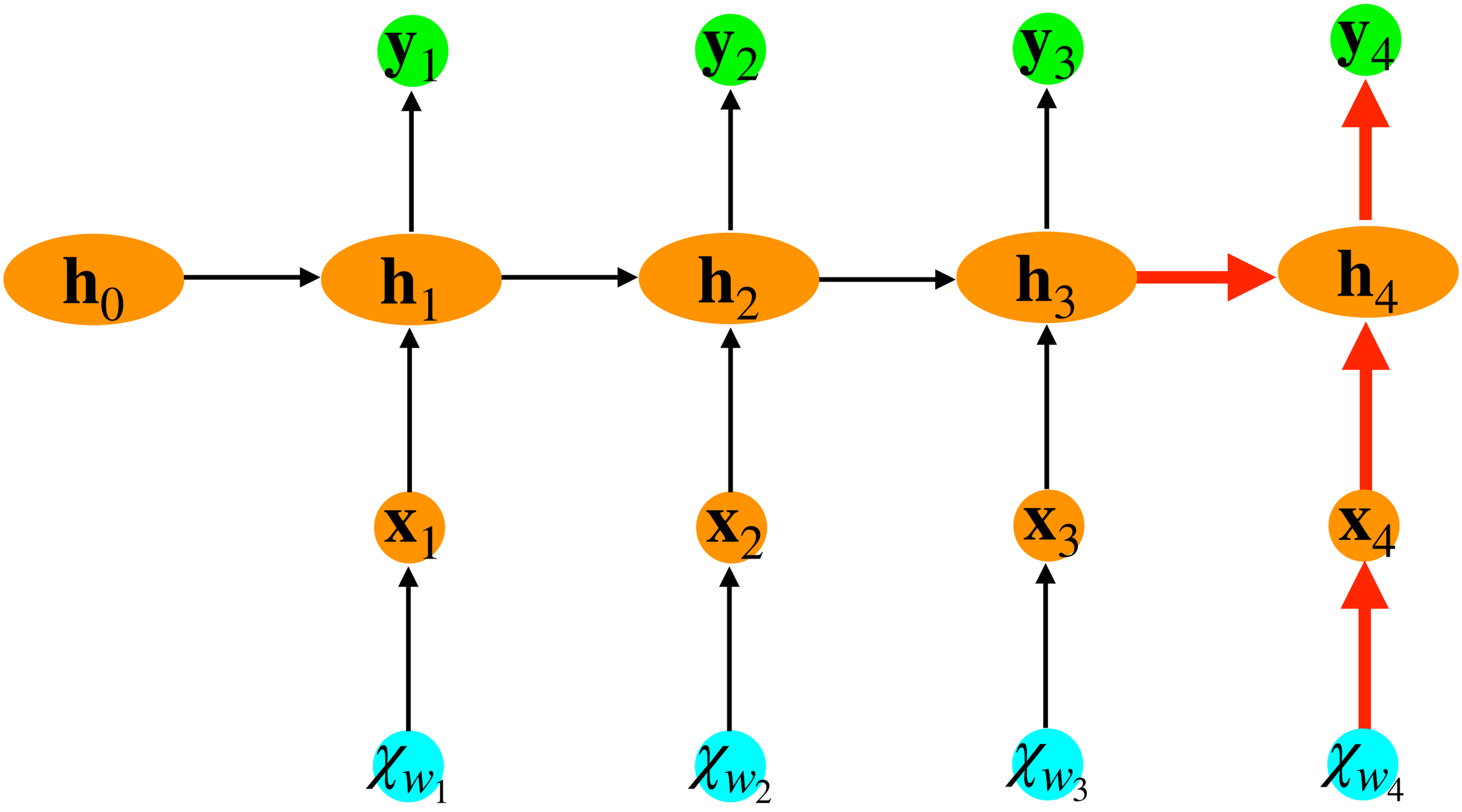
Пропагиране напред при рекурентна невронна мрежа



Пропагиране напред при рекурентна невронна мрежа



Пропагиране напред при рекурентна невронна мрежа



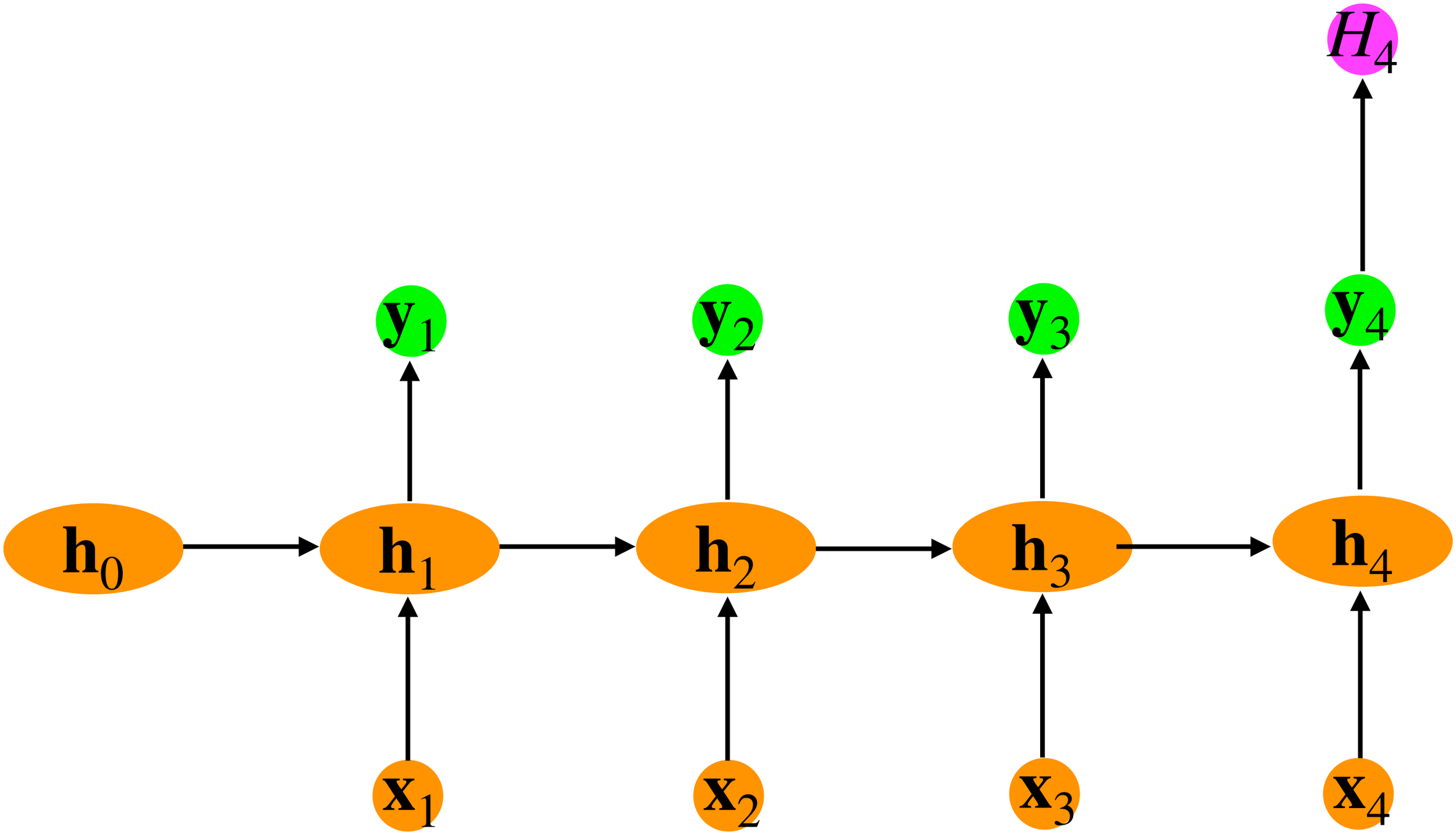
План на лекцията

1. Формалности за курса (5 мин)
2. Не-марковски рекурентен езиков модел (10 мин)
3. Пропагиране напред при рекурентна невронна мрежа (5 мин)
4. **Пропагиране назад при рекурентна невронна мрежа (5 мин)**
5. Особености при обучение на рекурентна невронна мрежа (30 мин)
6. Проблем и решение при експлодиращ градиент (10 мин)
7. Проблем и архитектури за решаване на проблема при изчезващ градиент (25 мин)

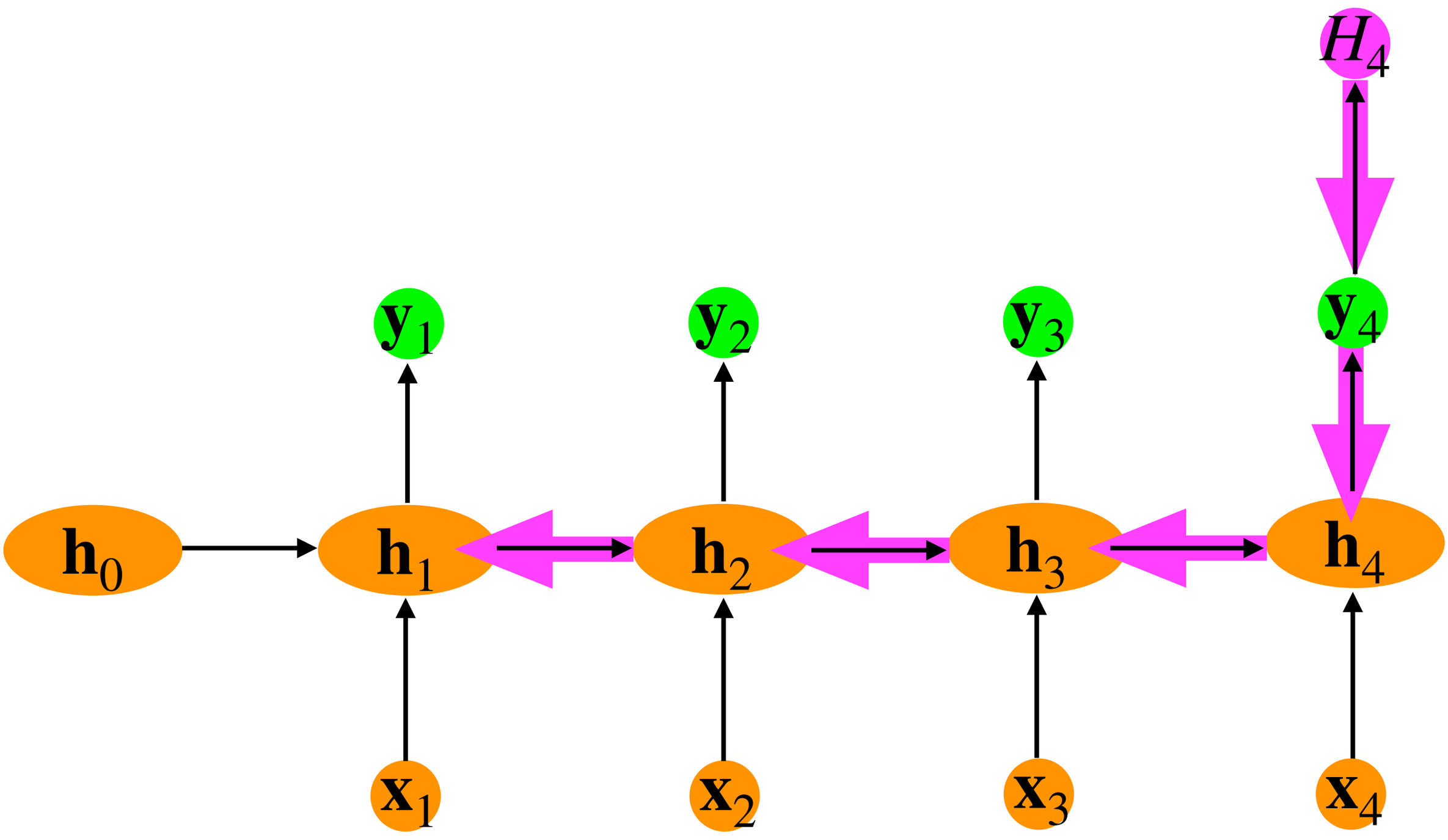
Пресмятане на градиентите с Backpropagation

- Трябва да съставим изчислителен граф
- Изчислителния граф следва да бъде разпънат за всички думи в дадено изречение
- Имайки разпънатия изчислителен граф изчисляването на градиентите става автоматично с метода Backpropagation
- Тази техника се нарича “Backpropagation in Time”

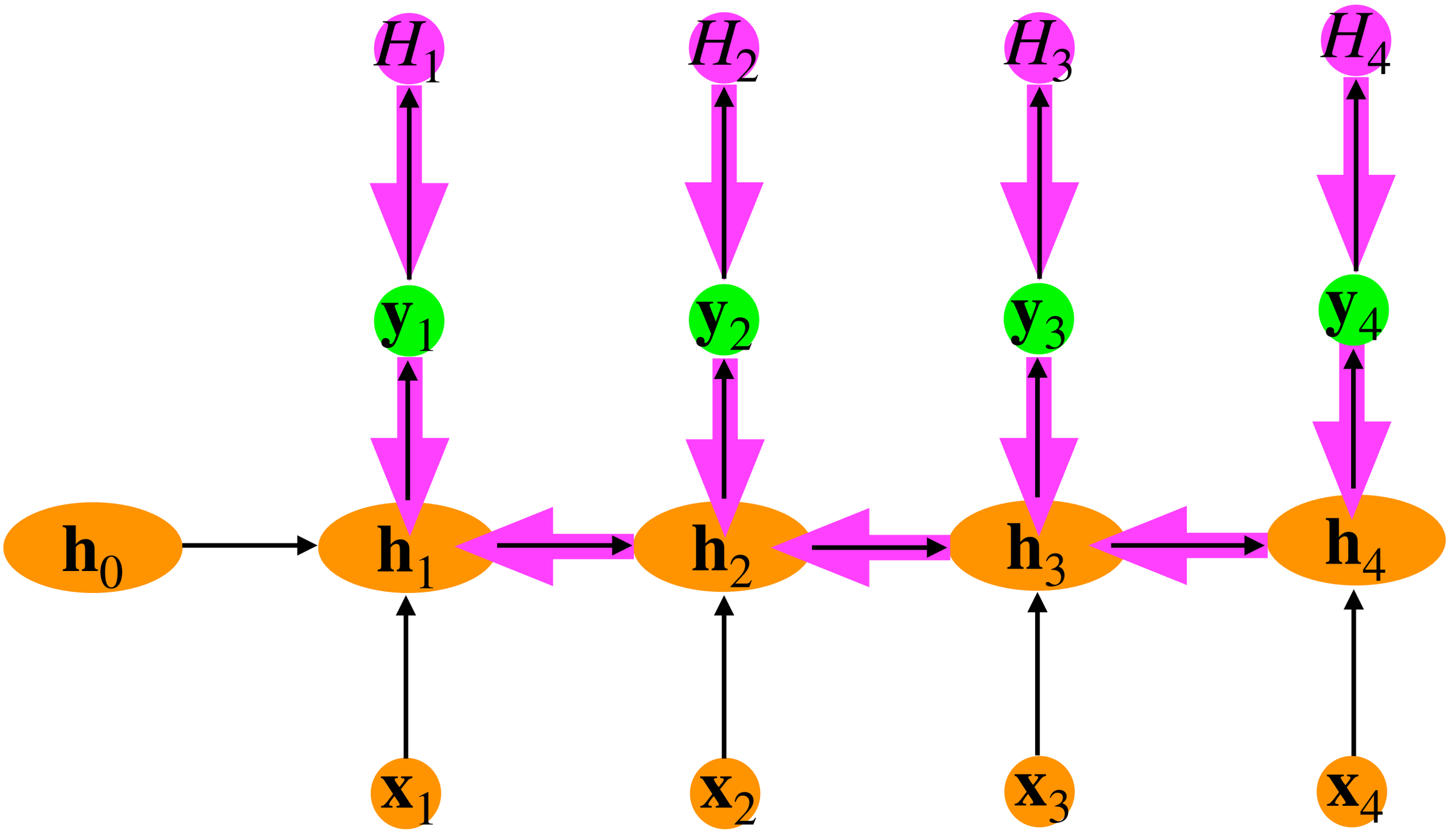
Пропагиране при рекурентни невронни мрежи



Пропагиране при рекурентни невронни мрежи



Пропагиране при рекурентни невронни мрежи



Партидно изчисляване на градиентите

- **Проблем**: Изреченията са с различна дължина.
- Решения:
  - Изравняваме всички изречения в дадена партида, като ги допълваме с нов специален символ, при който не пропагираме градиента назад.
  - Сортираме изреченията по дължина и ги групираме в партиди с еднаква дължина.
  - Конкатенираме изреченията и след това ги нарязваме на фиксирана дължина.

План на лекцията

1. Формалности за курса (5 мин)
2. Не-марковски рекурентен езиков модел (10 мин)
3. Пропагиране напред при рекурентна невронна мрежа (5 мин)
4. Пропагиране назад при рекурентна невронна мрежа (5 мин)
5. **Особености при обучение на рекурентна невронна мрежа (30 мин)**
6. Проблем и решение при експлодиращ градиент (10 мин)
7. Проблем и архитектури за решаване на проблема при изчезващ градиент (25 мин)

Обучение на рекурентна невронна мрежа

- Ще използваме минимизация на кросентропията, което е еквивалентно на максимизация на правдоподобието.

- Ще предполагаме, че ни е даден корпус $\mathbf{X}$. Елементите на корпуса са документи/изречения. С $w \in \mathbf{X}$ означаваме даден документ. С w_i означаваме номера на i-тия терм в документа w.

$$
\begin{aligned}
H_{\mathbf{X}}(E, V, W, U) &= -\frac{1}{\|\mathbf{X}\|} \sum_{w \in \mathbf{X}} \sum_{i=1}^{|w|} \log \Pr_{E,V,W,U}[w_{i+1} \mid w_1 w_2 \ldots w_i] \\
&= -\frac{1}{\|\mathbf{X}\|} \sum_{w \in \mathbf{X}} \sum_{i=1}^{|w|} \log(\mathbf{y}_i)_{w_{i+1}} \\
&= -\frac{1}{\|\mathbf{X}\|} \sum_{w \in \mathbf{X}} \sum_{i=1}^{|w|} \log \operatorname{softmax}(Ug(W\mathbf{h}_{i-1} + VE\chi_{w_i}))_{w_{i+1}}
\end{aligned}
$$

Обучение на рекурентна невронна мрежа

- Нека поточковата кросентропия в точката w_{i+1} означим с $H_{w_{i+1}}(E, V, W, U) = -\log(\mathbf{y}_i)_{w_{i+1}}$. Тогава:

$$
\begin{aligned}
H_{w_4} &= -\log \operatorname{softmax}(U\mathbf{h}_3)_{w_4} = \\
&= -\log \operatorname{softmax}(Ug(W\mathbf{h}_2 + VE\chi_{w_3}))_{w_4} = \\
&= -\log \operatorname{softmax}(Ug(Wg(W\mathbf{h}_1 + VE\chi_{w_2}) + VE\chi_{w_3}))_{w_4} = \\
&= -\log \operatorname{softmax}(Ug(Wg(Wg(W\mathbf{h}_0 + VE\chi_{w_1}) + VE\chi_{w_2}) + VE\chi_{w_3}))_{w_4}
\end{aligned}
$$

Ще трябва да намерим градиентите по параметрите.

- $\frac{\partial}{\partial \mathbf{t}} \log \text{softmax}(\mathbf{t})_k = (\bar{\delta}_k - \text{softmax}(\mathbf{t}))$
- $\frac{\partial H_{w_{i+1}}}{\partial U} = -\frac{\partial}{\partial U} \log \text{softmax}(U\mathbf{h}_i)_{w_{i+1}} = -(\bar{\delta}_{w_{i+1}} - \text{softmax}(U\mathbf{h}_i)) \otimes \mathbf{h}_i$
- $\frac{\partial H_{w_{i+1}}}{\partial W} = -\frac{\partial}{\partial W} \log \text{softmax}(U\mathbf{h}_i)_{w_{i+1}}$

- Нека положим $\mathbf{z}_i = W\mathbf{h}_{i-1} + VE\chi_{w_i}$. Тогава $\mathbf{h}_i = g(\mathbf{z}_i)$.
- Означения:
  - $g'(\mathbf{a})$ е диагонална матрица с диагонал $g'(\mathbf{a}_i)$.
  - Ако $A \in \mathbb{R}^{L \times M}$е матрица и $\mathbf{b} \in \mathbb{R}^N$ е вектор то $A \otimes \mathbf{b} \in \mathbb{R}^{L \times M \times N}$ и $(A \otimes \mathbf{b})_{i,j,k} = A_{i,j}\mathbf{b}_k$.
  - $\mathbf{I}_N \in \mathbb{R}^{N \times N}$ е единичната матрица.

От лекция 7

$$\frac{\partial}{\partial \mathbf{b}} \log \frac{e^{(W\mathbf{x}+\mathbf{b})_y}}{\sum_{j=1}^{K} e^{(W\mathbf{x}+\mathbf{b})_j}} = (\bar{\delta}_y - \text{softmax}(W\mathbf{x}+\mathbf{b}))^\top \left(\frac{\partial}{\partial \mathbf{b}} (W\mathbf{x}+\mathbf{b}) \right) =$$

$$= (\bar{\delta}_y - \text{softmax}(W\mathbf{x}+\mathbf{b}))^\top \mathbf{I} =$$

$$= \bar{\delta}_y - \text{softmax}(W\mathbf{x}+\mathbf{b})$$

Разглеждаме функцията: $u : \mathbb{R}^{KN} \to \mathbb{R}^K, u(W) = W\mathbf{x} + \mathbf{b}$. Якобиянът $\frac{\partial \mathbf{u}}{\partial W}$ е матрица $\mathbb{R}^{K \times KN}$.

$$\left(\frac{\partial \mathbf{u}}{\partial W_{p,q}} \right)_k = \frac{\partial \mathbf{u}_k}{\partial W_{p,q}} = \frac{\partial \sum_{l=1}^{N} W_{k,l} x_l}{\partial W_{p,q}} = \begin{cases} 0 & \text{if } k \neq p \\ x_q & \text{if } k = p \end{cases} = \delta_{p=k} x_q$$

$$\frac{\partial}{\partial W} \log \frac{e^{(W\mathbf{x}+\mathbf{b})_y}}{\sum_{j=1}^{K} e^{(W\mathbf{x}+\mathbf{b})_j}} = (\bar{\delta}_y - \text{softmax}(W\mathbf{x}+\mathbf{b}))^\top \left(\frac{\partial}{\partial W} (W\mathbf{x}+\mathbf{b}) \right) =$$

$$= (\bar{\delta}_y - \text{softmax}(W\mathbf{x}+\mathbf{b}))^\top \frac{\partial \mathbf{u}}{\partial W} =$$

$$= (\bar{\delta}_y - \text{softmax}(W\mathbf{x}+\mathbf{b})) \otimes \mathbf{x}$$

Защото, ако $\mathbf{v} = \bar{\delta}_y - \text{softmax}(W\mathbf{x}+\mathbf{b})$, то:

$$\left(\mathbf{v}^\top \frac{\partial \mathbf{u}}{\partial W} \right)_{p,q} = \mathbf{v}^\top \frac{\partial \mathbf{u}}{\partial W_{p,q}} = \sum_{k=1}^{K} \mathbf{v}_k \left(\frac{\partial \mathbf{u}}{\partial W_{p,q}} \right)_k = \sum_{k=1}^{K} \mathbf{v}_k \delta_{p=k} \mathbf{x}_q = \mathbf{v}_p \mathbf{x}_q$$

- $$\frac{\partial H_{w_{i+1}}}{\partial W} = \frac{\partial H_{w_{i+1}}}{\partial \mathbf{h}_i}\frac{\partial \mathbf{h}_i}{\partial W} = -\left(\bar{\delta}_{w_{i+1}} - \text{softmax}(U\mathbf{h}_i)\right)^\top U\frac{\partial \mathbf{h}_i}{\partial W}$$
- $$\frac{\partial \mathbf{h}_i}{\partial W} = \frac{\partial}{\partial W}g(W\mathbf{h}_{i-1} + VE\chi_{w_i}) = g'(\mathbf{z}_i)\left(W\frac{\partial \mathbf{h}_{i-1}}{\partial W} + \mathbf{I}_N \otimes \mathbf{h}_{i-1}\right)$$
- $$\frac{\partial H_{w_{i+1}}}{\partial V} = \frac{\partial H_{w_{i+1}}}{\partial \mathbf{h}_i}\frac{\partial \mathbf{h}_i}{\partial V} = -\left(\bar{\delta}_{w_{i+1}} - \text{softmax}(U\mathbf{h}_i)\right)^\top U\frac{\partial \mathbf{h}_i}{\partial V}$$
- $$\frac{\partial \mathbf{h}_i}{\partial V} = \frac{\partial}{\partial V}g(W\mathbf{h}_{i-1} + VE\chi_{w_i}) = g'(\mathbf{z}_i)\left(W\frac{\partial \mathbf{h}_{i-1}}{\partial V} + \mathbf{I}_N \otimes E\chi_{w_i}\right)$$
- $$\frac{\partial H_{w_{i+1}}}{\partial E} = \frac{\partial H_{w_{i+1}}}{\partial \mathbf{h}_i}\frac{\partial \mathbf{h}_i}{\partial E} = -\left(\bar{\delta}_{w_{i+1}} - \text{softmax}(U\mathbf{h}_i)\right)^\top U\frac{\partial \mathbf{h}_i}{\partial E}$$
- $$\frac{\partial \mathbf{h}_i}{\partial E} = \frac{\partial}{\partial E}g(W\mathbf{h}_{i-1} + VE\chi_{w_i}) = g'(\mathbf{z}_i)\left(W\frac{\partial \mathbf{h}_{i-1}}{\partial E} + V \otimes \chi_{w_i}\right)$$

$$
\begin{aligned}
-\frac{\partial H_{w_4}}{\partial W} &= (\bar{\delta}_{w_4} - \text{softmax}(U\mathbf{h}_3))^\top U \frac{\partial \mathbf{h}_3}{\partial W} = \\
&= (\bar{\delta}_{w_4} - \text{softmax}(U\mathbf{h}_3))^\top U g'(\mathbf{z}_3)\left(\mathbf{I}_N \otimes \mathbf{h}_2 + W\frac{\partial \mathbf{h}_2}{\partial W}\right) = \\
&= (\bar{\delta}_{w_4} - \text{softmax}(U\mathbf{h}_3))^\top U g'(\mathbf{z}_3)\left(\mathbf{I}_N \otimes \mathbf{h}_2 + W g'(\mathbf{z}_2)\left(\mathbf{I}_N \otimes \mathbf{h}_1 + W\frac{\partial \mathbf{h}_1}{\partial W}\right)\right) = \\
&= (\bar{\delta}_{w_4} - \text{softmax}(U\mathbf{h}_3))^\top U g'(\mathbf{z}_3)\left(\mathbf{I}_N \otimes \mathbf{h}_2 + W g'(\mathbf{z}_2)\left(\mathbf{I}_N \otimes \mathbf{h}_1 + W g'(\mathbf{z}_1)\left(\mathbf{I}_N \otimes \mathbf{h}_0 + W\frac{\partial \mathbf{h}_0}{\partial W}\right)\right)\right) = \\
&= (\bar{\delta}_{w_4} - \text{softmax}(U\mathbf{h}_3))^\top U g'(\mathbf{z}_3)\left(\mathbf{I}_N \otimes \mathbf{h}_2 + W g'(\mathbf{z}_2)\left(\mathbf{I}_N \otimes \mathbf{h}_1 + W g'(\mathbf{z}_1)\mathbf{I}_N \otimes \mathbf{h}_0\right)\right) = \\
&= (\bar{\delta}_{w_4} - \text{softmax}(U\mathbf{h}_3))^\top U \left(g'(\mathbf{z}_3)\mathbf{I}_N \otimes \mathbf{h}_2 + g'(\mathbf{z}_3) W g'(\mathbf{z}_2)\left(\mathbf{I}_N \otimes \mathbf{h}_1 + W g'(\mathbf{z}_1)\mathbf{I}_N \otimes \mathbf{h}_0\right)\right) = \\
&= (\bar{\delta}_{w_4} - \text{softmax}(U\mathbf{h}_3))^\top U \left(g'(\mathbf{z}_3)\mathbf{I}_N \otimes \mathbf{h}_2 + W g'(\mathbf{z}_3) g'(\mathbf{z}_2)\mathbf{I}_N \otimes \mathbf{h}_1 + W^2 g'(\mathbf{z}_3) g'(\mathbf{z}_2) g'(\mathbf{z}_1)\mathbf{I}_N \otimes \mathbf{h}_0\right)
\end{aligned}
$$

$$
\frac{\partial H_{w_{i+1}}}{\partial W} = -\left(\bar{\delta}_{w_{i+1}} - \text{softmax}(U\mathbf{h}_i)\right)^\top U g'(\mathbf{z}_i) \sum_{j=1}^{i} \left(\prod_{k=1}^{j-1} W g'(\mathbf{z}_{i-k})\right) \mathbf{I}_N \otimes \mathbf{h}_{i-j}
$$

По подобен начин се изразяват $\frac{\partial H_{w_{i+1}}}{\partial V}$ и $\frac{\partial H_{w_{i+1}}}{\partial E}$:

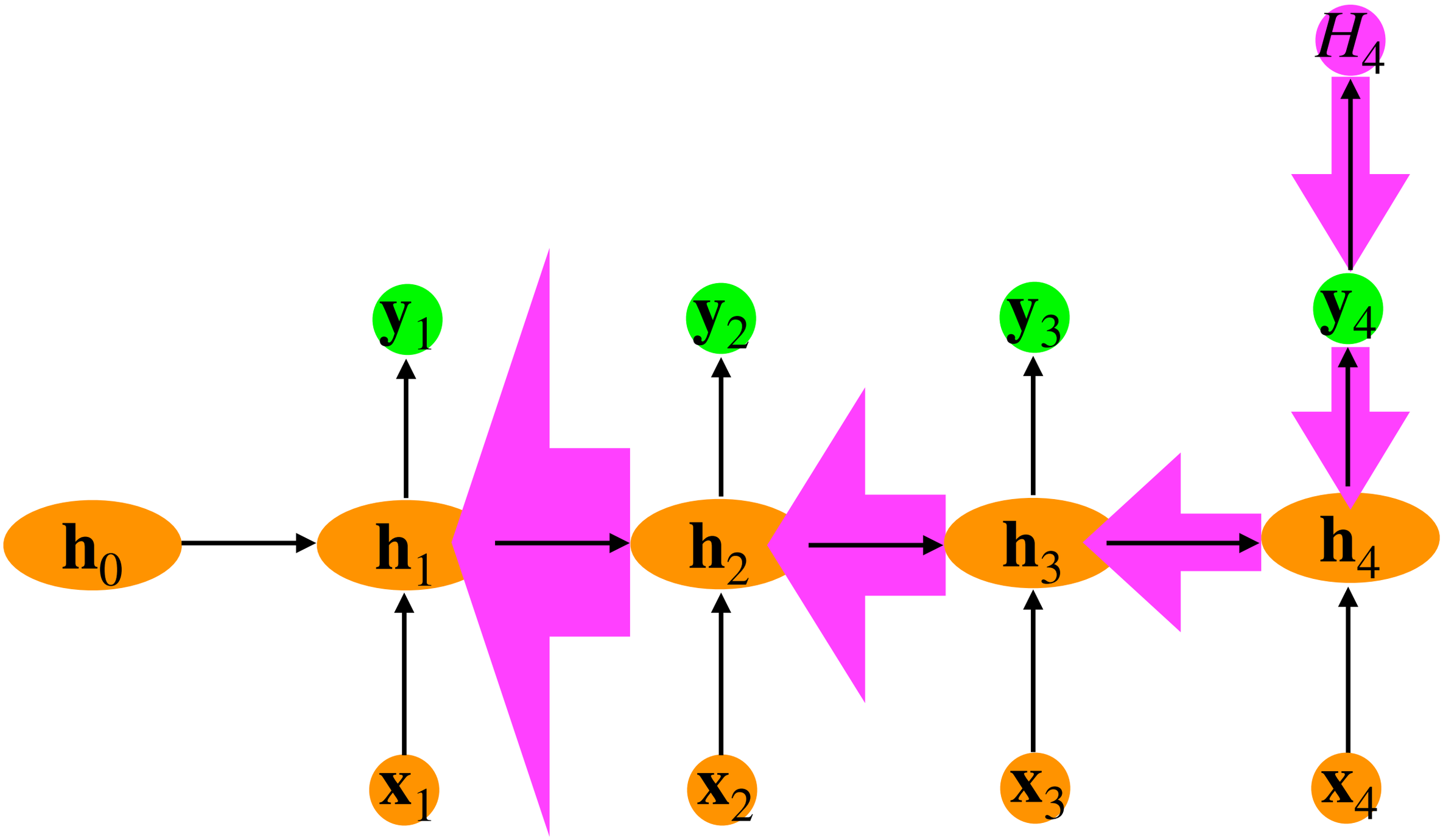
- $$\frac{\partial H_{w_{i+1}}}{\partial W} = -(\bar{\delta}_{w_{i+1}} - \text{softmax}(U\mathbf{h}_i))^\top U g'(\mathbf{z}_i) \sum_{j=1}^{i} \left(\prod_{k=1}^{j-1} W g'(\mathbf{z}_{i-k}) \right) \mathbf{I}_N \otimes \mathbf{h}_{i-j}$$
- $$\frac{\partial H_{w_{i+1}}}{\partial V} = -(\bar{\delta}_{w_{i+1}} - \text{softmax}(U\mathbf{h}_i))^\top U g'(\mathbf{z}_i) \sum_{j=1}^{i} \left(\prod_{k=1}^{j-1} W g'(\mathbf{z}_{i-k}) \right) \mathbf{I}_N \otimes E\chi_{i-j+1}$$
- $$\frac{\partial H_{w_{i+1}}}{\partial E} = -(\bar{\delta}_{w_{i+1}} - \text{softmax}(U\mathbf{h}_i))^\top U g'(\mathbf{z}_i) \sum_{j=1}^{i} \left(\prod_{k=1}^{j-1} W g'(\mathbf{z}_{i-k}) \right) V \otimes \chi_{i-j+1}$$
- Разглеждаме $\prod_{k=1}^{j-1} W g'(\mathbf{z}_{i-k})$ — съответства на градиента $\frac{\partial \mathbf{h}_i}{\partial \mathbf{h}_{i-j}}$.
- Как зависи нормата на градиента от разстоянието за пропагиране i ?

- Ако функцията $g(z) = \mathrm{ReLU}(z) = \max(0,z)$ то $\|g'(\mathbf{z})\| \leq 1$.
- Ако $\|W\| < 1$ то $\|Wg'(\mathbf{z}_{i-k})\| < 1$. Следователно градиента $\frac{\partial \mathbf{h}_i}{\partial \mathbf{h}_{i-j}} = \prod_{k=1}^{j-1} Wg'(\mathbf{z}_{i-k})$ намалява експоненциално — **ИЗЧЕЗВАЩ ГРАДИЕНТ**
- Ако $\|W\| > 1$ то градиента $\frac{\partial \mathbf{h}_i}{\partial \mathbf{h}_{i-j}} = \prod_{k=1}^{j-1} Wg'(\mathbf{z}_{i-k})$ евентуално може да расте експоненциално — **ЕКСПЛОДИРАЩ ГРАДИЕНТ**
- Ако функцията $g(z) = \sigma(z) = \frac{1}{1+e^{-z}}$ то $\|g'(\mathbf{z})\| \leq 1/4$ и за $\|W\| < 4$ или $\|W\| > 4$ получаваме съответно изчезващ или експлодиращ градиент

План на лекцията

1. Формалности за курса (5 мин)
2. Не-марковски рекурентен езиков модел (10 мин)
3. Пропагиране напред при рекурентна невронна мрежа (5 мин)
4. Пропагиране назад при рекурентна невронна мрежа (5 мин)
5. Особености при обучение на рекурентна невронна мрежа (30 мин)
6. **Проблем и решение при експлодиращ градиент (10 мин)**
7. Проблем и архитектури за решаване на проблема при изчезващ градиент (25 мин)

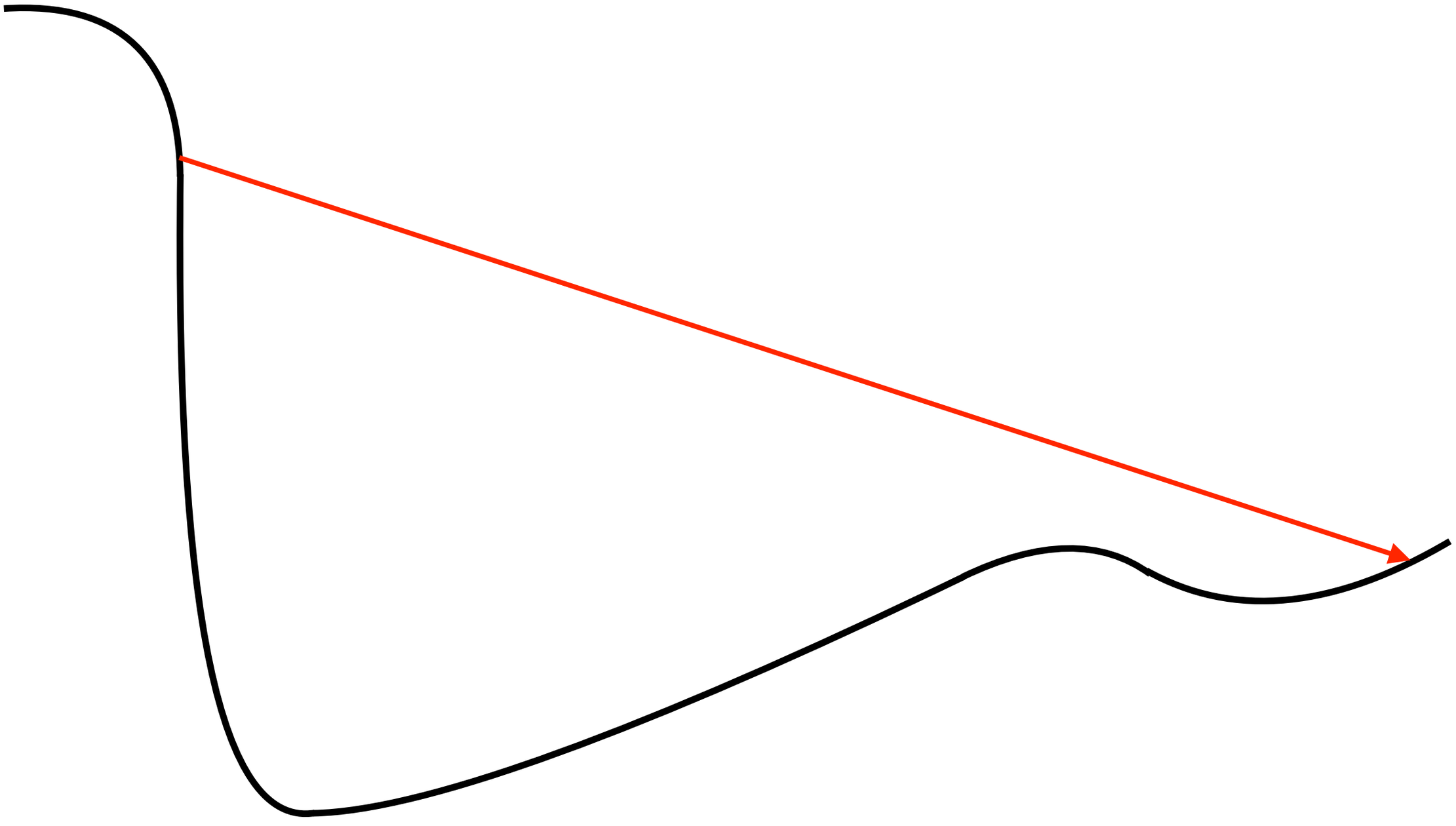
Пропагиране при експлодиращ градиент



Проблеми при експлодиращ градиент

- При спускане по градиента имаме:
  - $\theta_{k+1} = \theta_k - \alpha \nabla_\theta H(\theta_k)$
- Има опасност градиента да излезе извън обхвата на числата с плаваща запетая и да получим стойност **`Inf`** или **`NaN`**
- Ако градиента е много голям ще направим огромен скок при спускането по градиента

Проблеми при екплодиращ градиент



Решение: ограничаване на градиента — gradient clipping

- Ако нормата на градиента е над даден праг $\kappa > 0$, то преди да направим спускането намаляваме дължината на градиента до κ.

- По този начин ще направим спускане в същата посока но с по-малка стъпка:

- $\theta_{k+1} = \theta_k - \alpha \ \mathrm{clip}_\kappa(\nabla_\theta H(\theta_k))$, където

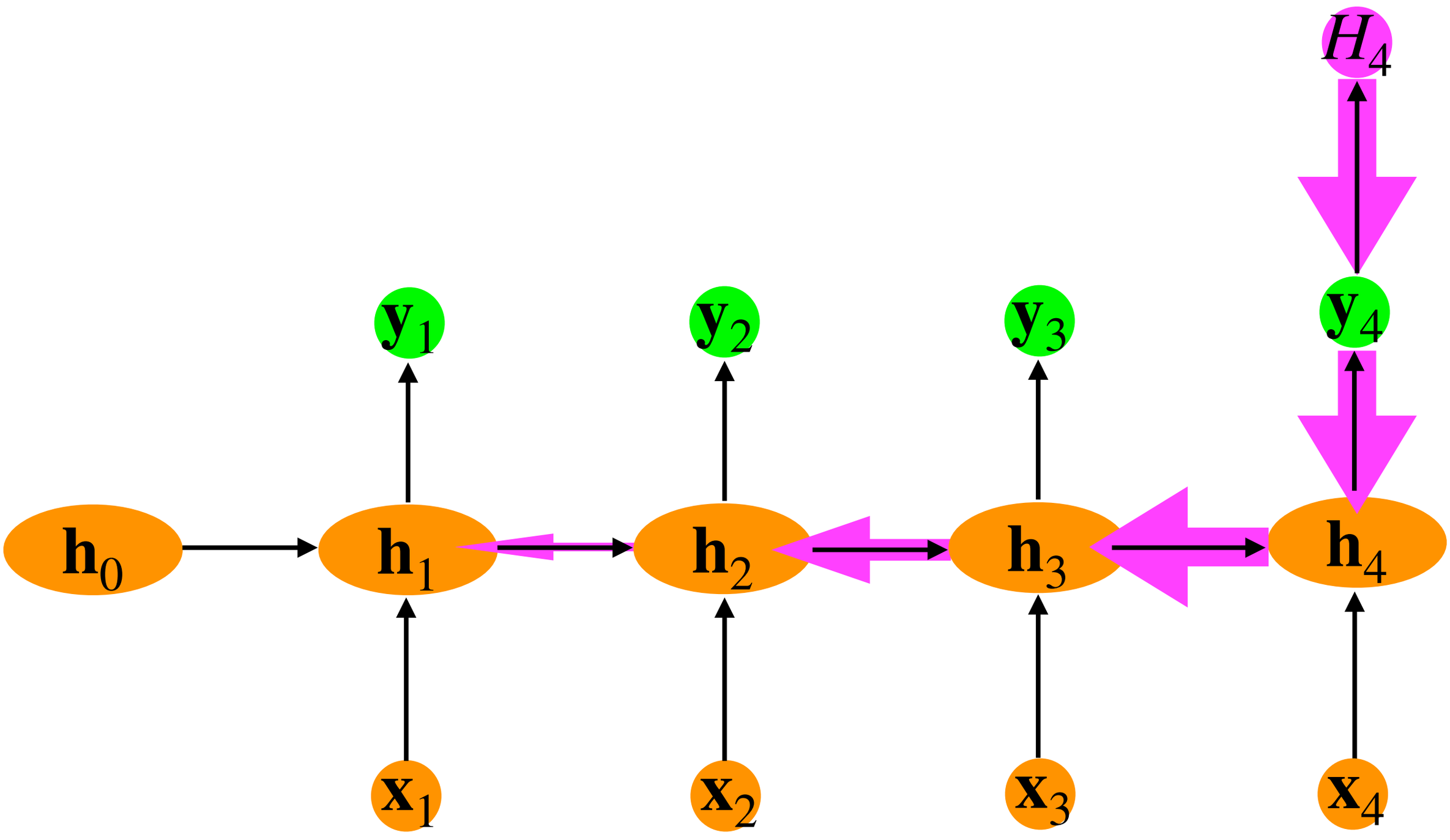
$$\mathrm{clip}_\kappa(\mathbf{u}) = \begin{cases} \frac{\kappa}{\|\mathbf{u}\|}\mathbf{u} & \text{if } \|\mathbf{u}\| > \kappa \\ \mathbf{u} & \text{if } \|\mathbf{u}\| \leq \kappa \end{cases}$$

- Решението е просто и се прилага често в дълбокото машинно обучение при най-различни невронни архитектури.

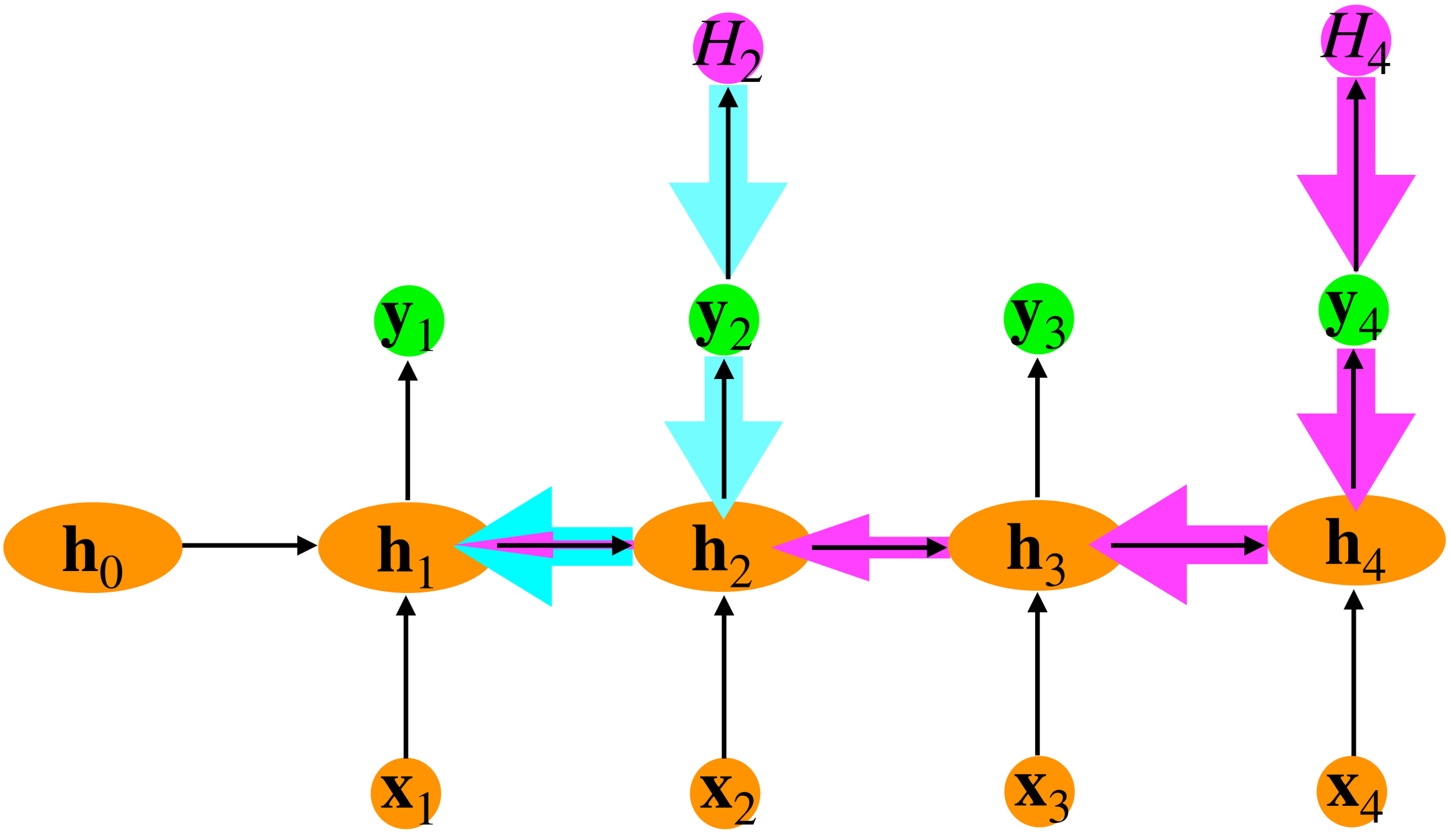
План на лекцията

1. Формалности за курса (5 мин)
2. Не-марковски рекурентен езиков модел (10 мин)
3. Пропагиране напред при рекурентна невронна мрежа (5 мин)
4. Пропагиране назад при рекурентна невронна мрежа (5 мин)
5. Особености при обучение на рекурентна невронна мрежа (30 мин)
6. Проблем и решение при експлодиращ градиент (10 мин)
7. **Проблем и архитектури за решаване на проблема при изчезващ градиент (25 мин)**

Пропагиране при рекурентни невронни мрежи



Пропагиране при рекурентни невронни мрежи близко и далечно разстояние — short term long term



Научаване на зависимости на голямо разстояние

- Пример:

  Книгата, която всички търсеха и толкова много харесваха, беше най-после върната от учителя по литература в библиотеката.

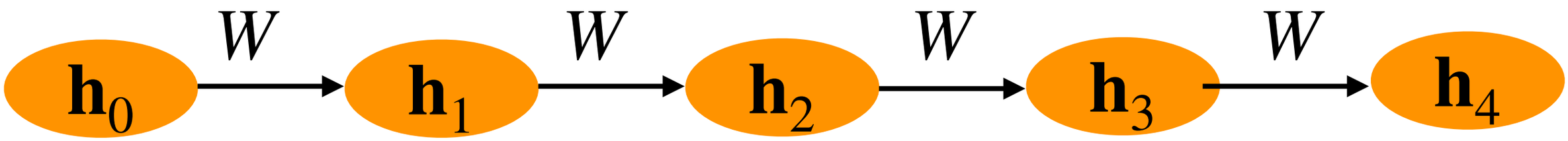
- Разстоянието между *книгата* и *библиотеката* е 15 думи.

- Моделът е желателно да научи зависимостта между *книгата* и *библиотеката*.

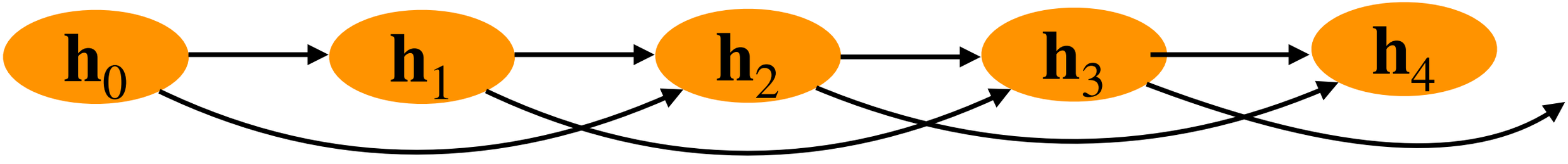
- Ако градиента през тези 15 позиции изчезне, моделът няма да може да научи тази зависимост.

Решаване на проблема с изчезващия градиент

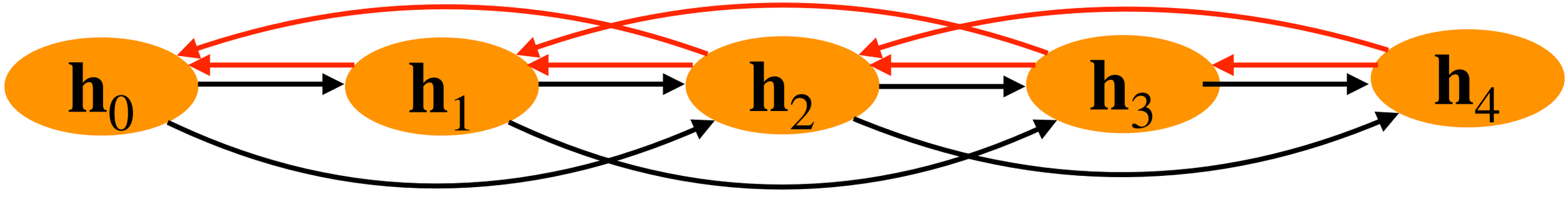
- Рекурентната формула $\mathbf{h}_i = g(W\mathbf{h}_{i-1} + V\mathbf{x}_i)$ води до вдигане на степен на матрицата W при пропагирането на градиента.



- Този проблем може да се реши, ако връзките между отделните състояния станат по-директни.



Решаване на проблема с изчезващия градиент

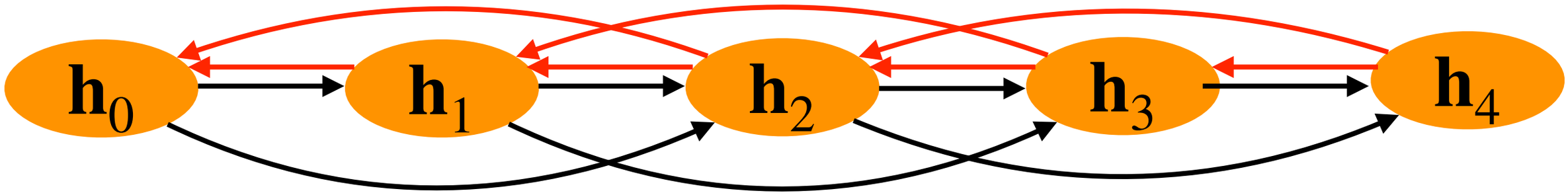


- Такива връзки наричаме **преки връзки** (shortcut connection, skip connection, residual connection).
  Подобни методи се използват в много от архитектурите с дълбоки невронни мрежи — skip-net, highway net, …

- През пряката връзка позволяваме на градиента да пропагира директно до предходните състояния.

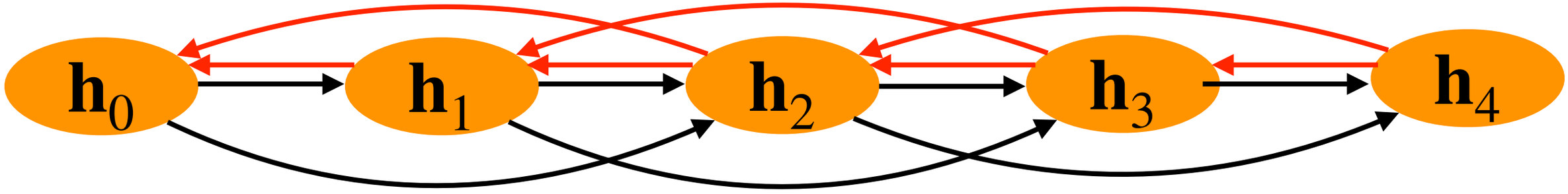
- Но за да се подобри обучението е целесъобразно да се контролира информацията по преките връзки.

Контрол на информацията с порти



- Нека приложим *адаптивни* преки връзки.
- $h_t = f(h_{t-1}, x_t) = u_t \odot \tilde{h}_t + (1 - u_t) \odot h_{t-1}$
  - кандидат за презапис: $\tilde{h}_t = \tanh(W[h_{t-1}; x_t] + b)$
  - **порта (gate)** за презапис: $u_t = \sigma(W_u[h_{t-1}; x_t] + b_u)$

Контрол на информацията с порти



- Нека позволим да премахнем *адаптивно* ненужни връзки.

- $h_t = u_t \odot \tilde{h}_t + (1 - u_t) \odot h_{t-1}$

  - кандидат за презапис: $\tilde{h}_t = \tanh(W[(r_t \odot h_{t-1}); x_t] + b)$

  - порта за презапис: $u_t = \sigma(W_u[h_{t-1}; x_t] + b_u)$

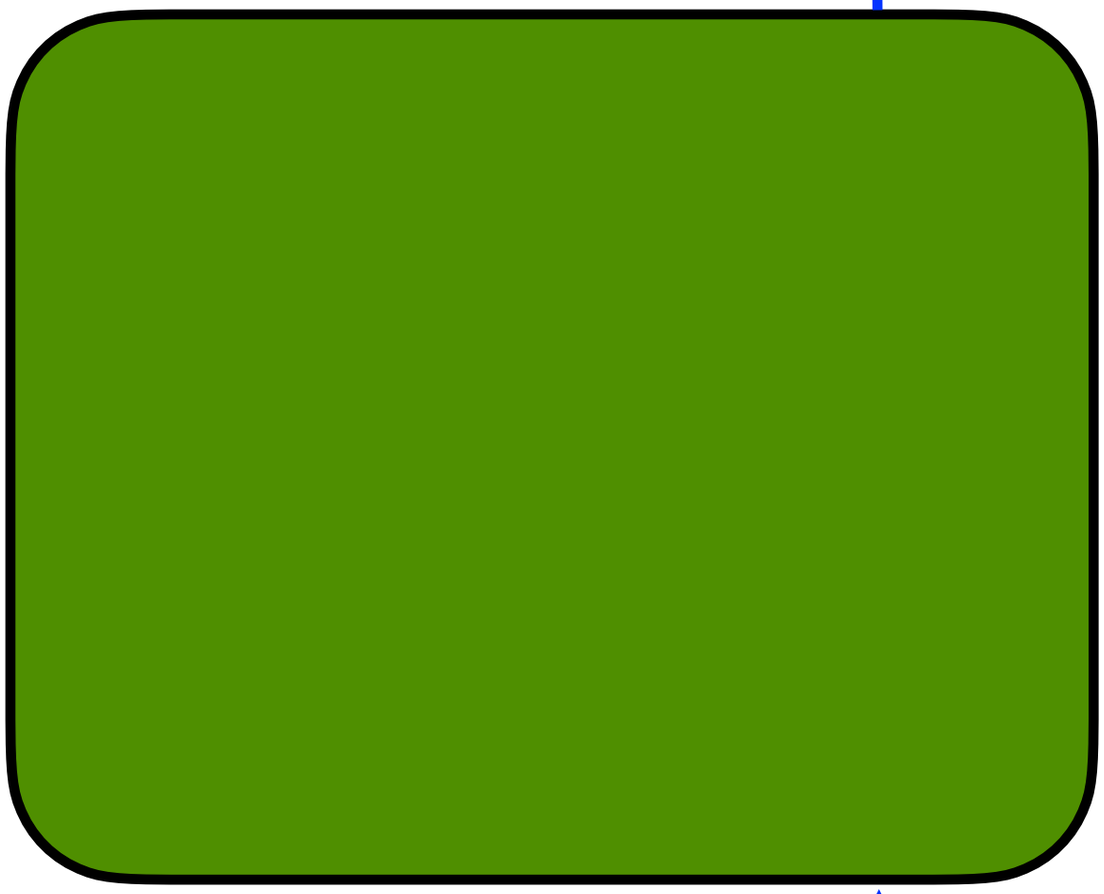
  - порта за нулиране: $r_t = \sigma(W_r[h_{t-1}; x_t] + b_r)$

Рекурентен елемент с порти
Gated recurrent unit GRU

- $h_t = u_t \odot \tilde{h}_t + (1 - u_t) \odot h_{t-1}$
- $\tilde{h}_t = \tanh(W[(r \odot h_{t-1}); x_t] + b)$
- $u_t = \sigma(W_u[h_{t-1}; x_t] + b_u)$
- $r_t = \sigma(W_r[h_{t-1}; x_t] + b_r)$
- $x_t \in \mathbb{R}^M, h_t, \tilde{h}_t, h_{t-1}, u_t, r_t \in \mathbb{R}^N$
- $W, W_u, W_r \in \mathbb{R}^{N \times (N+M)}, b, b_u, b_r \in \mathbb{R}^N$

Интуиция за GRU

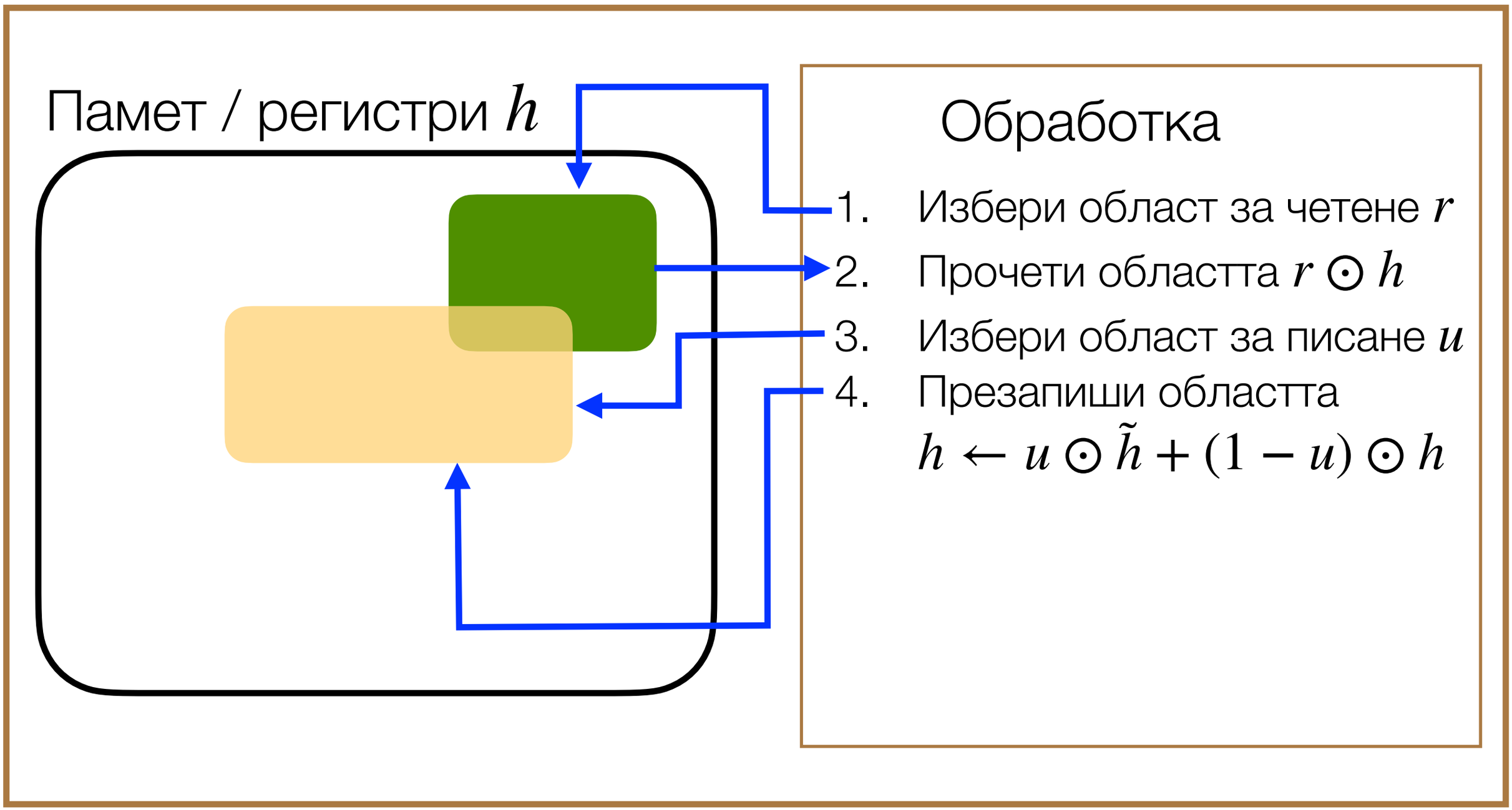
Памет / регистри h



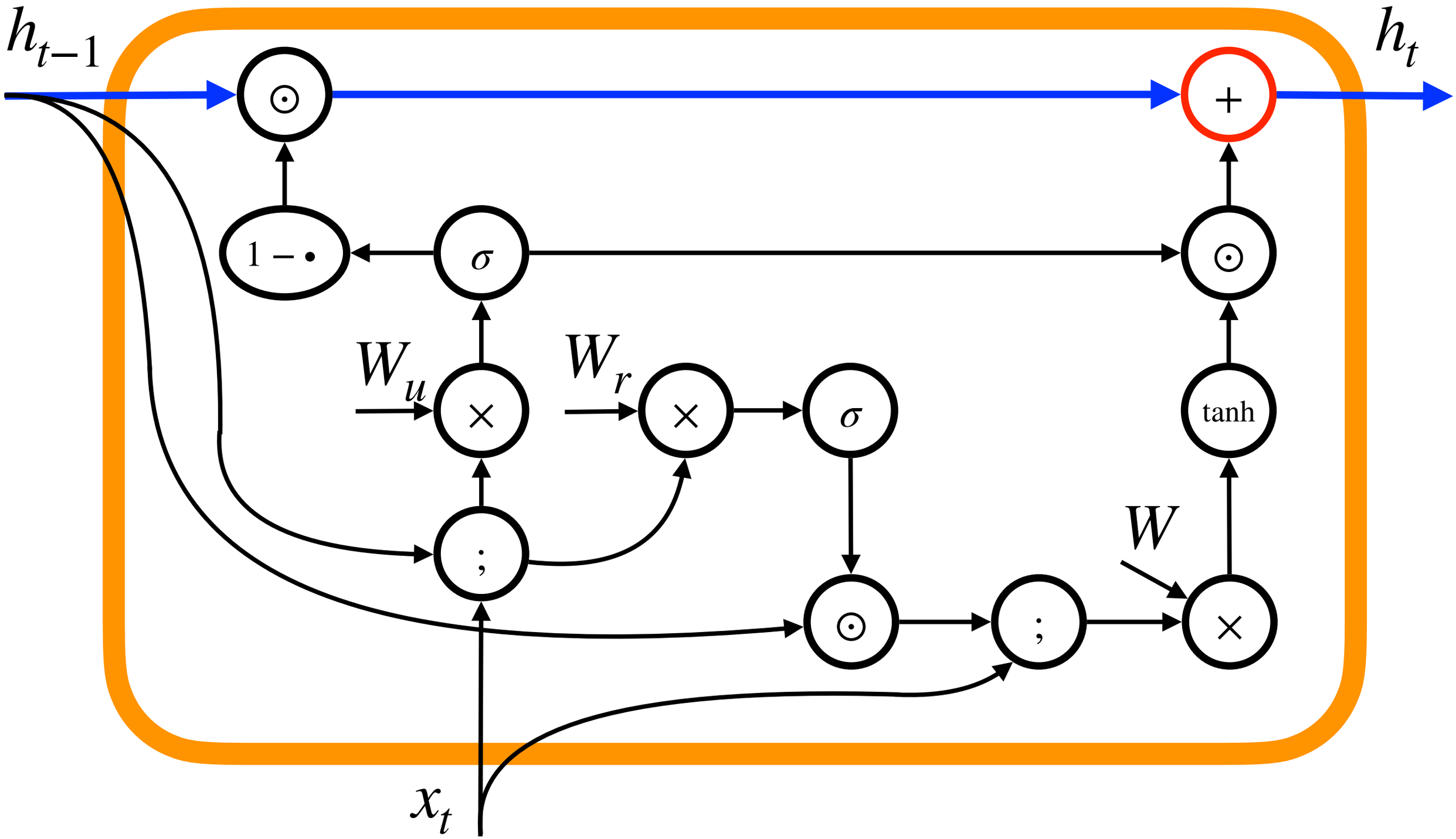
Обработка

1. Прочети цялата памет h
2. Презапиши цялата памет $h \leftarrow g(W[h; x] + b)$

Интуиция за GRU



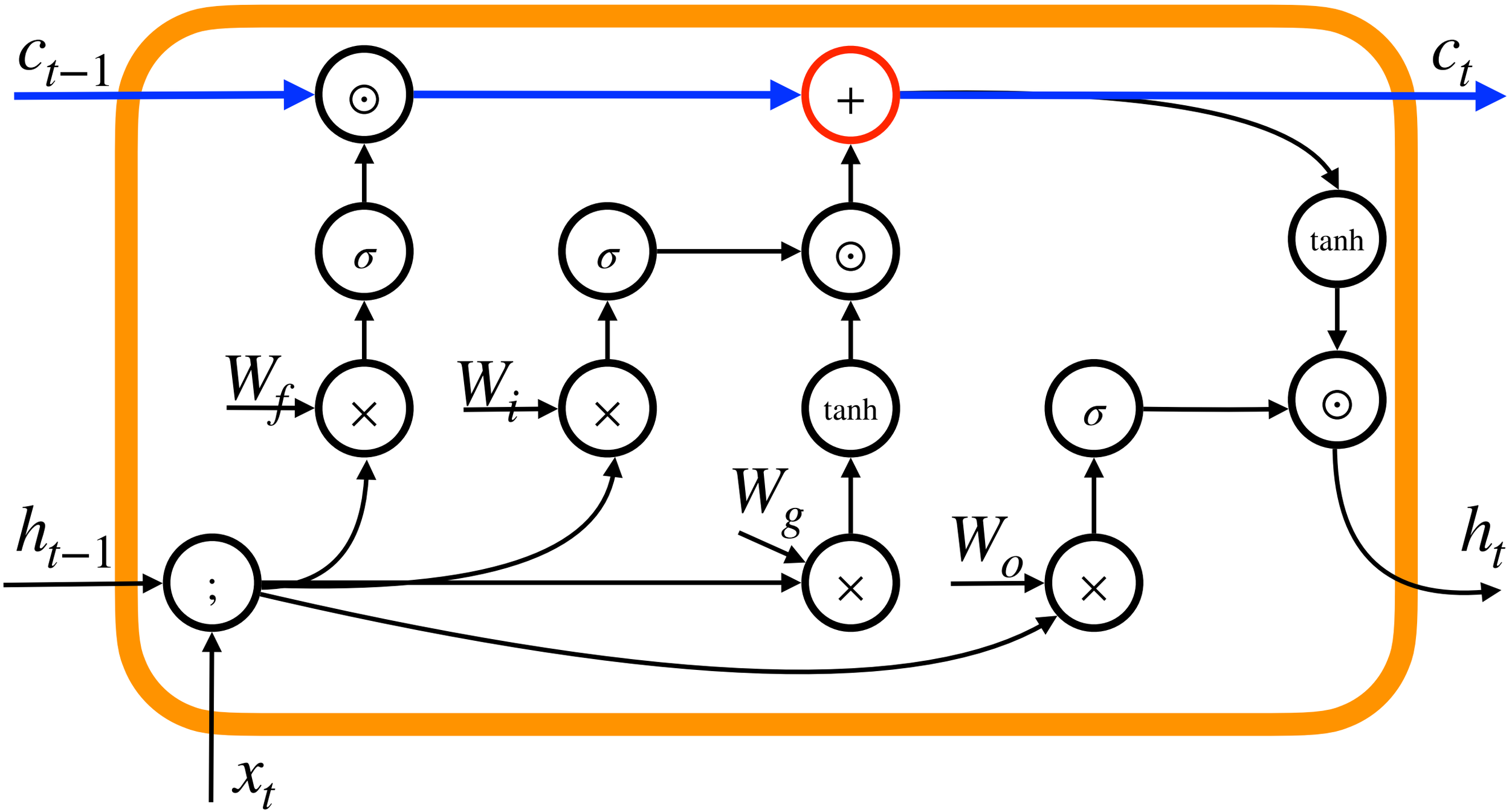
GRU изчислителен граф



Рекурентен елемент с краткосрочна и дългосрочна памет Long-short term memory LSTM

- $h_t = o_t \odot \tanh(c_t)$
- $c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$
- $\tilde{c}_t = \tanh(W_c[h_{t-1}; x_t] + b_c)$
- $o_t = \sigma(W_o[h_{t-1}; x_t] + b_o)$
- $i_t = \sigma(W_i[h_{t-1}; x_t] + b_i)$
- $f_t = \sigma(W_f[h_{t-1}; x_t] + b_f)$
- $x_t \in \mathbb{R}^M, h_t, c_t, \tilde{c}_t, c_{t-1}, o_t, i_t, f_t \in \mathbb{R}^N$
- $W_c, W_o, W_i, W_f \in \mathbb{R}^{N \times (N+M)}, b_c, b_o, b_i, b_f \in \mathbb{R}^N$

LSTM изчислителен граф



Сравнение между LSTM и GRU

- $h_t = o_t \odot \tanh(c_t)$
- $c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$
- $\tilde{c}_t = \tanh(W_c[h_{t-1}; x_t] + b_c)$
- $o_t = \sigma(W_o[h_{t-1}; x_t] + b_o)$
- $i_t = \sigma(W_i[h_{t-1}; x_t] + b_i)$
- $f_t = \sigma(W_f[h_{t-1}; x_t] + b_f)$

- $h_t = u_t \odot \tilde{h}_t + (1 - u_t) \odot h_{t-1}$
- $\tilde{h}_t = \tanh(W[(r \odot h_{t-1}); x_t] + b)$
- $u_t = \sigma(W_u[h_{t-1}; x_t] + b_u)$
- $r_t = \sigma(W_r[h_{t-1}; x_t] + b_r)$

Заключение

- РНН са универсална невронна архитектура за моделиране на езикови модели и контекстни зависимости
- LSTM и GRU са най-широко използваните в момента архитектури на рекурентни невронни мрежи
- Чрез използването на портали при тези архитектури се осъществява ефективно обучение на зависимости на голямо (потенциално неограничено) разстояние
- Всички съвременни платформи за дълбоки невронни мрежи имат готови оптимизирани имплементации на LSTM и GRU елементи