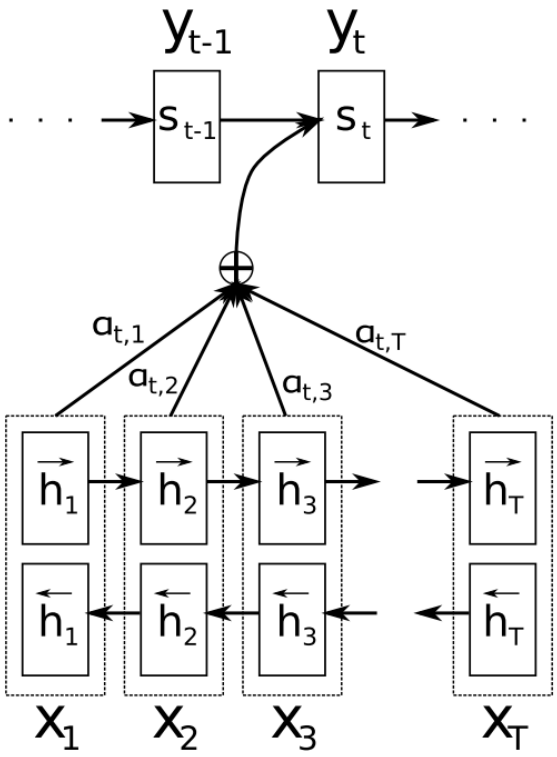
Ուշադրության մեխանիզմ

Հայկ Կարապետյան

Սովորական RNN օգտագործելիս առաջանում էր vanishing gradient խնդիր: LSTM և GRU բլոկները որոշ չափով լուծում էին այդ խնդիրը, բայց միևնույն է սկզբնական բառի մասին ինֆորմացիան (history) շատ ձևափոխությունների էր ենթարկվում: Այդ պատճառով անհրաժեշտ է կիրառել մի մեթոդ, որը առաջին բառի մշակման (թարգմանության) ժամանակ կհասկանա, թե որ բառերին պետք է ուշադրություն դարձնի: Առաջին բառի թարգմանության համար ցանցն է որոշելու, թե նախադասության որ բառերն են կարևոր և այդպես հերթով թագմանելու է նախադասությունը: Ամեն բառին ուշադրություն դարձնելու համար օգտագործում են ուշադրության (attention) մեխանիզմը: Attention-ի օրինակ կարող եք տեսնել նկար 1-ում:



Նկար 1: Attention-ի օրինակ

Հասկանանք, թե ինչ է պատկերված նկար 1-ում: Ներքևի շերտը bidirectional RNN է, որը մուտքում ստանում է հաջորդական տվյալներ ($x_1, \dots, x_T$) և վերադարձնում է հաջորդական տվյալներ ($h'_1, \dots h'_T$): Bidirectional RNN-ից դուրս եկող output-ը ամենաշատը ինֆորմացիա ունի իր ինդեքսով մուտքային տվյալի մասին, բայց նաև ինֆորմացիա ունի բոլոր տվյալների մասին: Օրինակ h'_2-ը ամենաշատ ինֆորմացիան ունի x_2 մասին, բայց նաև տեղյակ է $x_1, x_3, \dots x_T$ տվյալների մասին: Վերևին շերտում ունենք սովորական RNN, որը մուտքում ստանում է նույն $x_1, \dots, x_T$ տվյալները, բայց նաև ստանում է history ներքևի շերտից: Ամեն քայլի RNN-ին անհրաժեշտ է history ներքևի շերտից, որպեսզի միացնենք այս շերտի history-ի հետ և փոխնցենք հաջորդ բլոկին: c-ով նշանակենք ներքևի շերտից եկող history-ն: c-ն ստացվում է հետևյալ կերպ:

$$c_t = \alpha_{t1} * h'_1 + \alpha_{t2} * h'_2 + \dots + \alpha_{tT} * h'_T$$

α-ները 0-ից մեկ արժեքներ են և դրանց գումարը հավասար է 1-ի: Այսինքն ամեն α ցույց է տալիս տվյալը ինչքան կարևոր է history-ի մեջ: Այդ կարևորությունը որոշելու է նեյրոնային ցանցը ուսուցման ընթացքում: Այսինքն α-ն ուսուցանվող պարամետր է: α-ները ամեն history-ի

դեպքում տարբեր են և դա կարող է խնդիր առաջացնել: Երբ մուտքային տվյալների քանակը T է կունենանք T հատ α, իսկ երբ թեստավորման ժամանակ ունենանք T+3 հաջորդական տվյալ, այսինքն T+3 բառից կազմված նախադասություն, այդ դեպքում չենք կարող իրենց համար ուսուցանել համապատասխան α-ներ: Այս պատճառով անհրաժեշտ է որոշել α-ների ստացման մեթոդ, որը անկախ մուտքային տվյալների քանակից կաշխատի:
Այսպիսով ստացանք.

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j, \ i = 1, \ldots, T_y$$

α_{ij}-երի գումարը պետք է մեկ լինի:

Այդպես կարող ենք ստանալ օգտագործելով softmax ակտիվացիոն ֆունկցիա:

$$\alpha_{ij} = \frac{exp\{e_{ij}\}}{\sum_{j=1}^{T_y} exp\{e_i j\}}$$

Մնում է հայտարարել e_{ij}-ն

$$e_{ij} = F(s_{i-1}, h'_j) = W_1^T tanh(W_2 s_{i-1} + W_3 h'_j)$$

Նկար 1-ում վերևի RNN շերտը ուներ history, որին միանում էր ներքևի շերտից եկող c-ն: Ամեն α-ն ստանալու համար մեզ անհրաժեշտ է վերևի RNN-ի s_{i-1} history-ն և ներքևի շերտի h'_j-ն: Եթե մեր մուտքային հաջորդական տվյալների քանակը ավելի շատ լինի T-ից, այդ դեպքում h'-ի և s-ի քանակը նույնպես կշատանա, որովհետև սովորական RNN-ում մեր բլոկերի քանակը համապատասխանում էր մուտքային տվյալների քանակին (բոլոր բլոկերը ունեն նույն կշիռները): Եվ արդյունքում $x_{T+1}, x_{T+2}, x_{T+3}$ տվյալների համար նույնպես կունենանք α գործակիցներ: W_1, W_2 և W_3 կշիռները թարմացվում են ուսուցանման ընթացքում և մնում են նույնը թեստավորման ժամանակ: