Chương 6

Điều phối hệ thống thông minh

Các này đề cập những gì cần thiết để *vận hành* hệ thống thông minh. Điều này bao gồm việc kiểm soát tất cả các bộ phận của nó, giữ cho trí thông minh và kinh nghiệm của nó được cân bằng, phát triển nó, sửa lỗi và đảm bảo rằng nó liên tục đạt được các mục tiêu của mình. Nó thảo luận về các công cụ cần thiết, cách hiểu những lỗi mà hệ thống thông minh mắc phải và những việc cần làm để ứng phó cũng như những việc cần làm nếu mọi người bắt đầu lạm dụng hệ thống.

6.1. Tổng quan về điều phối thông minh

Sự phối hợp thông minh hơi giống một cuộc đua ô tô. Cả một nhóm người chế tạo một chiếc ô tô, đưa tất cả công nghệ mới nhất vào đó và thiết lập mọi cánh khí động học, chấn lưu, tỷ số truyền và van nạp một cách hoàn hảo; họ tạo ra một cỗ máy tuyệt vời có thể làm được những điều mà không cỗ máy nào khác có thể làm được.

Hình. Hậu cần đua xe ô tô.

Và sau đó ai đó cần phải ngồi sau tay lái, đưa nó vào đường đua và giành chiến thắng! Những người điều phối thông minh là những người điều khiển. Họ nắm quyền kiểm soát hệ thống thông minh và làm những gì cần thiết để giúp hệ thống đạt được mục tiêu. Họ sử dụng các hệ thống quản lý và sáng tạo trí thông minh để tạo ra trí thông minh phù hợp vào đúng thời điểm và kết hợp nó theo những cách hữu ích nhất. Họ điều khiển hệ thống đo

từ xa, thu thập dữ liệu cần thiết để cải thiện hệ thống thông minh. Và họ giải quyết mọi vấn đề sai lầm, cân bằng mọi thứ để hệ thống thông minh tạo ra giá trị cao nhất có thể cho người dùng và doanh nghiệp. Ví dụ:

- Nếu trí thông minh trở nên tốt hơn, trải nghiệm có thể trở nên mạnh mẽ hơn để đạt được kết quả tốt hơn.
- Nếu vấn đề thay đổi, việc quản lý thông tin có thể cần được tối ưu hóa để tạo và triển khai thông tin theo cách khác hoặc nhanh hơn.
- Nếu hệ thống không đạt được mục tiêu, ai đó cần tìm ra lý do và cách sử dụng các hệ thống sẵn có để thích ứng.

Xây dựng hệ thống thông minh và điều phối chúng là những hoạt động rất khác nhau. Họ đòi hỏi những tư duy rất khác nhau. Và cả hai đều cực kỳ quan trọng để đạt được thành công.

Cần mô tả một trí thông minh được tổ chức tốt trông như thế nào và giới thiệu các yếu tố của việc điều phối một hệ thống thông minh, bao gồm vòng đời, môi trường điều phối, các lỗi và sự lạm dụng.

6.1.1. Đặc tính của một trí thông minh được tổ chức tốt

Một hệ thống thông minh được tổ chức tốt sẽ:

1. *Đạt được các mục tiêu một cách đáng tin cậy theo thời gian*: Nghĩa là, nó sẽ phát triển, đạt được các mục tiêu ngày càng tốt hơn cho đến khi đạt đến điểm lợi nhuận giảm dần. Sau đó, nó sẽ ở đó, tạo ra giá trị khi cơ sở người dùng, vấn đề và thế giới rộng lớn hơn thay đổi.
2. *Có kinh nghiệm, trí thông minh và mục tiêu cân bằng*: Nghĩa là kinh nghiệm sẽ có sức mạnh như trí thông minh hỗ trợ nên lợi ích ròng cho người dùng và doanh nghiệp được tối đa hóa.
3. *Giảm thiểu sai sót một cách hiệu quả*: Để những sai sót gây tốn kém được hiểu rõ, được phát hiện nhanh chóng và được sửa chữa. Và do đó, những sai sót về chi phí thấp hơn sẽ được giảm thiểu và không gây gánh nặng quá lớn cho người dùng hoặc doanh nghiệp.
4. *Mở rộng quy mô một cách hiệu quả theo thời gian*: Để chi phí duy trì hệ thống ở trạng thái tốt (tất cả việc cân bằng, tạo thông tin, giảm thiểu sai sót…) sẽ tăng theo số lượng người dùng và

độ phức tạp của thang đo thông tin. Đặc biệt, điều này có nghĩa là học máy và tự động hóa được sử dụng một cách hiệu quả.

5. *Xuống cấp chậm*: Tức là không cần tốn nhiều công sức để giữ ở trạng thái ổn định. Nỗ lực điều phối cần tập trung vào việc tìm kiếm cơ hội chứ không nên tranh giành để giữ cho bánh xe tiếp tục hoạt động.

6.1.2. Lí do cần phối hợp

Việc điều phối trí thông minh bao gồm việc điều chỉnh nó sao cho nó tạo ra giá trị cao nhất có thể trong suốt vòng đời của nó.

Học máy có nhiệm vụ điều chỉnh hệ thống trong suốt vòng đời của nó. Cái này là cái gì? Một trò đùa nào đó? Tiếc là không có. Trí tuệ nhân tạo và học máy sẽ chỉ giúp tiến xa hơn. Điều phối là sử dụng những công cụ đó và đặt chúng vào những tình huống tốt nhất để chúng có thể tạo ra giá trị, làm nổi bật điểm mạnh, bù đắp điểm yếu, và phản ứng khi mọi thứ thay đổi theo thời gian.

Sự phối hợp có thể cần thiết vì:

- Những thay đổi mục tiêu của hệ thống thông minh.
- Người dùng của hệ thống thông minh thay đổi.
- Vấn đề thay đổi.
- Trí tuệ thay đổi.
- Chi phí vận hành hệ thống thông minh thay đổi.
- Ai đó cố gắng lạm dụng hệ thống thông minh.

Một hoặc nhiều điều này gần như chắc chắn sẽ xảy ra trong vòng đời của hệ thống thông minh. Bằng cách học cách xác định chúng và thích nghi, có thể biến những vấn đề tiềm ẩn này thành cơ hội.

6.1.2.1. Thay đổi mục tiêu

Khi mục tiêu thay đổi, cách tiếp cận cũng sẽ phải thay đổi và điều đó có nghĩa là phải cân bằng lại các bộ phận trong hệ thống thông minh. Có thể muốn thay đổi mục tiêu của mình khi bất kỳ điều nào sau đây xảy ra:

- *Hiểu vấn đề tốt hơn*: Khi làm việc gì đó, sẽ hiểu nó rõ hơn. có thể nhận ra rằng ngay từ đầu đã đặt sai mục tiêu và muốn điều chỉnh. Nhớ lại rằng ở đó thường có nhiều lớp mục tiêu. có thể có một mục tiêu kinh doanh hoàn toàn tốt đẹp, nhưng hãy hiểu rằng kết quả

của người dùng mà đang theo dõi không góp phần vào mục tiêu; hoặc có thể nhận ra rằng các thuộc tính mô hình mà đang tối ưu hóa không đưa đi đúng hướng. Trong những trường hợp này, thay đổi mục tiêu và điều chỉnh hệ thống khi cần thiết để đạt được các mục tiêu mới.

- *Đã giải quyết được vấn đề trước đó*: có thể thấy rằng mình đã đạt được điều mình mong muốn. hệ thống thông minh đang chạy, người dùng nhận được kết quả tốt, doanh nghiệp hoạt động tốt. Tại thời điểm này, có thể muốn đặt mục tiêu cao hơn, và điều chỉnh hệ thống cho phù hợp. Hoặc có thể muốn tìm ra cách vận hành hệ thống với chi phí rẻ hơn (điều này sẽ thảo luận sau).
- *Nhận ra vấn đề trước đó quá khó giải quyết*: Đôi khi phải thừa nhận thất bại. có thể có một hệ thống đã chạy được một thời gian nhưng vẫn chưa đạt được mục tiêu mong muốn. Có thể dừng lại và tìm việc khác để làm hoặc có thể đặt mục tiêu dễ dàng hơn, thay đổi cách tiếp cận và xây dựng chậm hơn.
- *Khám phá được một số cơ hội mới*: có thể nảy ra một ý tưởng tuyệt vời. Một cái gì đó tương tự như hệ thống thông minh hiện tại, nhưng chỉ khác một chút. Có lẽ một trải nghiệm hoàn toàn mới có thể thúc đẩy trí thông minh. Có thể có thể xây dựng một loại trí thông minh mới dựa trên phương pháp đo từ xa hiện có. Đôi khi việc theo đuổi một cơ hội mới sẽ cần đến hệ thống thông minh mới, nhưng đôi khi điều đó có thể được thực hiện bằng những hệ thống đã có, bằng cách thay đổi mục tiêu.

6.1.2.2. Người dùng thay đổi

Cơ sở người dùng của hệ thống thông minh sẽ thay đổi theo vòng đời của nó theo nhiều cách khác nhau:

- *Người dùng mới đến*: Và người dùng mới có thể rất khác so với người dùng hiện tại. Ví dụ: có thể thu hút đối tượng nhân khẩu học mới, những người dùng bình thường hơn, những người dùng sử dụng các phần khác nhau của hệ thống thông minh hoặc những người dùng có suy nghĩ khác. Khi người dùng thay đổi, sẽ có cơ hội học hỏi và thích nghi. cũng sẽ có nhiều phương pháp đo từ xa hơn để sử dụng nhằm tạo ra thông tin thông minh và sẽ phải giải

quyết các vấn đề lấy mẫu ngày càng phức tạp cũng như độ phức tạp trong quản lý thông tin.

- *Mô hình sử dụng thay đổi*: Khi người dùng tìm hiểu cách sử dụng hệ thống thông minh, họ có thể thay đổi cách họ tương tác với nó. Họ sẽ ngừng khám phá cách thức hoạt động của nó và tập trung hơn vào việc thu được giá trị từ nó. Người dùng có kinh nghiệm sẽ được hưởng lợi từ các tương tác khác với người dùng mới làm quen và do đó, có thể muốn hệ thống thông minh của mình thay đổi khi người dùng trở thành chuyên gia.
- *Người dùng thay đổi nhận thức của họ về trải nghi*ệm: Theo thời gian, người dùng có thể trở nên mệt mỏi với những trải nghiệm thường xuyên và mạnh mẽ. Họ có thể bắt đầu phớt lờ chúng. Họ có thể ngày càng khó chịu. Một trải nghiệm hoạt động tốt trong tháng đầu tiên sử dụng có thể hoàn toàn sai về lâu dài, và có thể cần phải thay đổi.
- *Người dùng rời đi*: Và khi họ rời đi, có thể sẽ buồn một chút. Nhưng cuộc sống sẽ tiếp tục. Tùy thuộc vào việc người dùng nào rời đi, có thể có cơ hội thay đổi sự cân bằng giữa trí thông minh và trải nghiệm để tối ưu hóa cho người dùng. Cũng sẽ có ít khả năng đo từ xa hơn, điều này có thể làm giảm khả năng tạo ra trí thông minh. Đặc biệt, trí thông minh tự động có thể cần phải được xem xét lại; ví dụ, để tận dụng nhiều dữ liệu cũ hơn thay vì dựa vào dữ liệu mới hơn.

6.1.2.3. Vấn đề thay đổi

Hệ thống thông minh dành cho những vấn đề lớn, khó, có kết thúc mở và thay đổi theo thời gian. Những vấn đề này, theo định nghĩa, thay đổi theo thời gian:

- *Các vấn đề thay đổi theo thời gian luôn thay đổi*: Và do đó, những cách tiếp cận và quyết định đưa ra trong quá khứ có thể không phù hợp với tương lai. Đôi khi một vấn đề có thể dễ dàng. Vào những lúc khác nó có thể sẽ rất khó khăn. Đôi khi sẽ thấy rất nhiều loại bối cảnh cụ thể. Vào những lúc khác, sẽ không nhìn thấy những bối cảnh đó. Khi một vấn đề thay đổi, hầu như luôn có cơ hội để thích ứng và đạt được kết quả tốt hơn thông qua việc điều phối.

- *Mô hình sử dụng thay đổi*: Khi người dùng thay đổi cách họ tương tác với hệ thống thông minh, các vấn đề cần giải quyết có thể trở nên rất khác nhau. Ví dụ: người dùng hành xử khác nhau trong mùa hè hơn là vào mùa đông; vào cuối tuần họ cư xử khác với ngày thường; họ tương tác với nội dung mới khác với cách họ tương tác với nội dung cũ.
- *Bắt đầu giải quyết vấn đề*: Trong đó trí thông minh bắt đầu bắt kịp một vấn đề lớn, khó. Trong trường hợp này có thể muốn xem lại cách tiếp cận. Có thể muốn bắt đầu khóa hành vi thay vì tiếp tục phát triển, sử dụng nhiều bộ nhớ đệm thông minh hơn (bảng tra cứu) và đầu tư ít hơn vào đo từ xa.

6.1.2.4. Thay đổi thông minh

Trí thông minh của hệ thống sẽ thay đổi khi thực hiện bất kỳ thao tác nào sau đây:

- Thu thập nhiều dữ liệu hơn để xây dựng trí thông minh tốt hơn: Khi người dùng tương tác với hệ thống, trí thông minh sẽ có nhiều dữ liệu hơn và sẽ hoạt động tốt hơn. Khi trí thông minh tốt hơn, trải nghiệm có thể trở nên mạnh mẽ hơn.
- Hãy thử một cách tiếp cận mới giúp mọi việc tốt hơn rất nhiều: Dữ liệu mở ra những khả năng. Một số kỹ thuật học máy mạnh mẽ nhất không hiệu quả với dữ liệu *nhỏ*, nhưng trở nên khả thi khi người dùng đến với hệ thống thông minh và nhận được rất nhiều dữ liệu. Những loại thay đổi này có thể mở ra mọi loại tiềm năng để thử những trải nghiệm mới hoặc nhắm tới những mục tiêu táo bạo hơn.
- Hãy thử một cách tiếp cận mới khiến mọi việc trở nên tồi tệ hơn: Nhưng sẽ không hoàn hảo. Đôi khi sự phát triển trí thông minh đi ngược lại. Đây có thể là một vấn đề cụ thể với trí thông minh được kết hợp một cách chặt chẽ. Nó cũng có thể là một vấn đề khi dựa vào nhiều sự can thiệp của con người vào việc tạo ra trí thông minh (ví dụ như với phương pháp phỏng đoán); đôi khi sự phức tạp vượt quá điểm bùng phát và một kế hoạch tạo ra trí thông minh dễ vỡ không thể theo kịp. Trong những trường hợp này, có thể cần phải dừng các phần khác của hệ thống để có thời gian giải quyết vấn đề.

6.1.2.5. Thay đổi chi phí

Các hệ thống lớn sẽ liên tục cần cân bằng giữa chi phí và giá trị. Những thay đổi để tiết kiệm tiền có thể đòi hỏi sự thỏa hiệp ở những nơi khác. Nói cách khác, có thể thay đổi trải nghiệm hoặc trí thông minh của mình theo những cách tiết kiệm được nhiều tiền trong khi chỉ giảm một chút giá trị đối với người dùng hoặc doanh nghiệp. Có thể cố gắng giảm một hoặc cả hai điều sau:

1. *Chi phí đo từ xa*: Bằng cách thay đổi cách lấy mẫu dữ liệu. Điều này làm giảm khả năng đo lường hệ thống và tạo ra thông tin thông minh, nhưng đó có thể là một sự đánh đổi tốt. Ví dụ: có thể thu thập ít dữ liệu hơn khi trí thông minh không phát triển nhiều, lấy ít mẫu hơn từ các phần của vấn đề đã được giải quyết hoặc loại bỏ các phần bối cảnh không đóng góp vào các đặc trưng hữu ích (và các mô hình tốt hơn).
2. *Chi phí sai lầm*: Quản lý sai lầm có thể rất tốn kém. Có thể thấy rằng mình cần phạm ít sai lầm hơn, ít tốn kém hơn và có thể cần phải thay đổi tất cả mọi thứ trong hệ thống để thực hiện việc này.

6.1.2.6. Lạm dụng

Internet tràn ngập những trò trêu đùa. Một số người trong số họ lạm dụng dịch vụ vì họ cho rằng điều đó thật thú vị. Hầu hết họ lạm dụng dịch vụ (và người dùng) để kiếm tiền, hoặc khiến khó kiếm tiền hơn. Nếu không được giải quyết, việc lạm dụng có thể phá hỏng hệ thống thông minh, khiến nó trở thành nơi chứa thư rác và có nguy cơ khiến người dùng từ bỏ nó.

Để bảo vệ hệ thống thông minh khỏi bị lạm dụng, cần hiểu những kẻ lạm dụng muốn gì và sau đó cần làm điều gì đó để khiến hệ thống kém thú vị hơn đối với những kẻ lạm dụng tiềm năng. Đôi khi một tinh chỉnh nhỏ sẽ làm được điều đó. Đôi khi việc chống lạm dụng sẽ trở thành một phần quan trọng trong việc vận hành hệ thống thông minh.

Nhưng lạm dụng là một điều khó khăn. Khi một kẻ lạm dụng tìm ra cách kiếm được tiền cho mỗi lần tương tác từ người dùng, họ sẽ mở rộng quy mô. Họ sẽ nói với bè của họ. Sự lạm dụng có thể biểu hiện nhanh chóng và dữ dội, vì vậy cần chuẩn bị để nhận biết và phản ứng.

6.1.3. Điều phối

Việc điều phối một hệ thống thông minh đòi hỏi một bộ kỹ năng đa dạng. Đặc biệt, nhóm nên:

- Trở thành chuyên gia về lĩnh vực kinh doanh hệ thống thông minh, để họ hiểu mục tiêu theo bản năng và biết người dùng muốn gì khi tương tác với hệ thống.
- Hiểu trải nghiệm và có khả năng xem xét các tương tác cũng như quyết định cách cải thiện cách trình bày thông tin cho người dung; điều gì dễ, điều gì khó, người dùng sẽ thích điều gì, và những gì họ sẽ không làm.
- Hiểu cách thực hiện để họ biết cách phát hiện các vấn đề và có khả năng cải thiện việc thực hiện khi cần thiết.
- Có thể đặt tất cả các loại câu hỏi về dữ liệu, hiểu và truyền đạt câu trả lời.
- Biết cách áp dụng học máy để có thể tạo ra trí tuệ mới và đưa vào hệ thống khi cần thiết.
- Nhận được sự hài lòng từ việc làm cho hệ thống hoạt động hiệu quả hàng ngày. Không phải ai cũng thích điều này; một số người thích phát minh ra những thứ mới. Điều phối là sự xuất sắc trong việc thực hiện, thực hiện một số trong số những cỗ máy tuyệt vời nhất trên thế giới, đưa chúng vào công việc.

Một nhóm điều phối có thể bao gồm một số ít người làm đủ mọi nghề. Hoặc nó có thể bao gồm những người tham gia có kỹ năng đa dạng và làm việc hiệu quả cùng nhau.

6.1.4. Tóm tắt

Hệ thống thông minh cần được điều phối trong suốt vòng đời của nó để thành công. Điều phối là quá trình giữ cho tất cả các bộ phận của hệ thống làm tốt những gì chúng giỏi và hỗ trợ chúng khi chúng gặp vấn đề. Một hệ thống thông minh được tổ chức tốt sẽ thực hiện tất cả những điều sau

- Đạt được mục tiêu một cách đáng tin cậy theo thời gian.
- Có kinh nghiệm, trí tuệ và khách quan một cách cân bằng.
- Giảm thiểu sai sót một cách hiệu quả.
- Mở rộng quy mô một cách hiệu quả theo thời gian.
- Suy thoái chậm.

Một hoạt động quan trọng của việc điều phối là tái cân bằng trí thông minh và trải nghiệm trong vòng đời của hệ thống thông minh. Ví dụ, làm cho trải nghiệm trở nên mạnh mẽ hơn khi trí thông minh được cải thiện; hoặc làm cho nó bớt mạnh mẽ hơn khi vấn đề trở nên khó khăn hơn.

Việc điều phối cũng liên quan đến việc xử lý sai sót, hiểu rõ chúng là gì và khi nào chúng xảy ra, sau đó quản lý hệ thống để làm cho chúng ít tốn kém hơn. Những lý do cần có sự phối hợp bao gồm:

- Mục tiêu thay đổi.
- Người dùng thay đổi.
- Vấn đề thay đổi.
- Trí tuệ thay đổi.
- Hệ thống cần có chi phí vận hành rẻ hơn.

Đối thủ và kẻ lạm dụng thường trở thành vấn đề đối với các hệ thống thông minh thành công. Họ phải được xác định và thuyết phục rằng hệ thống không đáng để họ dành thời gian.

6.2. Môi trường điều phối thông minh

Dưới đây thảo luận về một số hoạt động chung góp phần tạo nên thành công trong việc điều phối thông tin thông minh. Đây là những ví dụ về những thứ có thể giúp tận dụng tối đa hệ thống thông minh và bao gồm những điều sau:

- *Giám sát tiêu chí thành công*: Để biết liệu hệ thống có đạt được mục tiêu hay không và nó đang trở nên tốt hơn hay tệ hơn.
- *Kiểm tra tương tác*: Để trải nghiệm các bối cảnh như người dùng đã trải nghiệm và xem kết quả mà người dùng đạt được.
- *Cân bằng trải nghiệ*m: Để làm cho trải nghiệm trở nên mạnh mẽ hơn hoặc ít hơn, có thể thay đổi loại tương tác hoặc có thể thay đổi mức độ nổi bật và tần suất của các tương tác.
- *Ghi đè thông tin*: Để sửa các lỗi xấu không thường xuyên hoặc giúp tối ưu hóa các ngữ cảnh phổ biến.
- *Tạo trí thông minh mới*: Quản lý trí thông minh và tạo ra các mô hình mới để giải quyết các vấn đề mới nổi như vấn đề, người dùng hoặc phạm vi thay đổi dữ liệu.

Hệ thống thông minh có thể hưởng lợi từ tất cả các hoạt động này và hơn thế nữa. Một số trong số chúng có thể được xây dựng như một phần của quá trình triển khai; ví dụ: bảng thông tin số liệu hiển thị tiến trình đạt được các mục tiêu tổng thể. Một số trong số chúng có thể được xử lý theo cách đặc biệt hơn; ví dụ: truy vấn các nguồn đo từ xa thô để ghép các tương tác đang gây ra sự cố lại với nhau.

Dưới đây lần lượt thảo luận về các hoạt động điều phối và đưa ra một số ý tưởng về cách có thể đầu tư vào từng lĩnh vực (nếu chọn), cùng với một số tiêu chí để quyết định đầu tư bao nhiêu.

6.2.1. Giám sát các tiêu chí thành công

Cần biết liệu hệ thống có đạt được mục tiêu hay không. Điều này liên quan đến việc thực hiện đo từ xa, điều chỉnh mọi trường hợp lấy mẫu và mất điện, đồng thời đưa ra câu trả lời mà tất cả những người tham gia đều có thể hiểu được; có thể là một vài con số, có thể là một biểu đồ. Hãy nhớ rằng điều này có thể liên quan đến bất kỳ điều nào sau đây:

- *Mục tiêu kinh doanh và các chỉ số dẫn đầu*, đặc biệt cho thấy chúng khác nhau như thế nào giữa những người dùng sử dụng nhiều phần thông minh của hệ thống so với những người không sử dụng.
- *Kết quả của người dùng*, cho biết hệ thống thông minh có hiệu quả như thế nào trong việc giúp người dùng đạt được kết quả mà mong muốn.
- *Các thuộc tính của mô hình*, cho biết tần suất thông tin là đúng hay sai (bất kể kết quả cuối cùng mà người dùng đạt được là gì). Giám sát các tiêu chí thành công là rất quan trọng. Mọi hệ thống thông minh chức năng đều phải có những người biết nó hoạt động tốt như thế nào hàng ngày và quan tâm đến việc nó đạt được mục tiêu đề ra.

Mức đầu tư cho việc giám sát các tiêu chí thành công bao gồm:

1. *Đặc biệt*: Khi một số người có kỹ năng đo lường các mục tiêu của hệ thống và đưa ra câu trả lời, có thể bằng cách thực hiện một số kiểm tra chéo các cài đặt đo từ xa hiện tại, điều chỉnh tập lệnh và chạy nó, sao chép vào biểu mẫu trình bày (như biểu đồ), và phân phối.

2. *Dựa trên công cụ*: Nơi mà bất kỳ ai trong nhóm đều có thể chạy một công cụ tạo và sau đó chạy truy vấn chính xác và đưa ra câu trả lời ở định dạng trình bày phù hợp.
3. *Tự động*: hệ thống tự động chạy các truy vấn một cách thường xuyên và lưu trữ kết quả ở nơi mà tất cả người tham gia có thể tìm thấy.
4. *Dựa trên cảnh báo*: trong đó hệ thống tự động hóa các số liệu nhưng cũng giám sát chúng và cung cấp cảnh báo cho người tham gia khi có điều gì đó xảy ra, bao gồm (i) *xuống cấp nghiêm trọng*: Khi các số liệu kém hơn nhiều so với lần đo trước đó. Ví dụ: số lỗi đã tăng 10% kể từ ngày hôm qua; (ii) xuống cấp đáng kể và kéo dài: Khi các số liệu có xu hướng giảm dần đủ lâu để vượt qua một ngưỡng nào đó. Ví dụ: tỷ lệ người dùng nhận được kết quả tốt đã giảm 10% trong tháng trước.
5. *Theo dõi lực lượng*: Nơi hệ thống theo dõi số liệu của các nhóm dân số quan trọng cũng như tổng dân số. Điều này có thể bao gồm những người từ một vị trí cụ thể, với cách sử dụng cụ thể, với nhân khẩu học cụ thể…

Tiêu chí đầu tư giám sát thành công: Hầu hết mọi hệ thống thông minh đều phải triển khai giám sát dựa trên cảnh báo đối với các tiêu chí thành công và các yếu tố quan trọng góp phần tạo nên tiêu chí thành công.

6.2.2. Kiểm tra tương tác

Kiểm tra tương tác bao gồm việc thu thập tất cả dữ liệu đo từ xa liên quan đến tương tác người dùng cụ thể và có thể xem tương tác từ đầu đến cuối. Điều này bao gồm bối cảnh của người dùng tại thời điểm tương tác, phiên bản của thông tin đang chạy; câu trả lời được đưa ra từ thông tin thông minh, trải nghiệm có được từ câu trả lời thông minh này, cách người dùng tương tác với trải nghiệm và kết quả cuối cùng mà người dùng nhận được. Điều này rất quan trọng để gỡ lỗi, theo dõi lỗi, để hiểu cách người dùng cảm nhận hệ thống thông minh và để có được trực giác về các tùy chọn trải nghiệm người dùng.

Mức đầu tư cho việc kiểm tra tương tác bao gồm

1. *Đặc biệt*: Nơi một số người có kỹ năng tìm ra tất cả các phần của sự tương tác và hiểu chúng liên quan như thế nào. Họ có thể phải

thực hiện một số công việc thám tử để xác định một tương tác cụ thể (hoặc các tương tác với một thuộc tính nhất định) và, dựa trên việc lấy mẫu, có thể hoặc không thể tìm thấy bất kỳ tương tác cụ thể nào.

2. *Dựa trên công cụ*: Nơi mà bất kỳ ai trong nhóm đều có thể xác định được sự tương tác (có thể theo người dùng và thời gian) xem dữ liệu đo từ xa có liên quan. Công cụ này cũng có thể hỗ trợ truy vấn các loại tương tác cụ thể (có thể ở đâu trí thông minh đưa ra một câu trả lời cụ thể và người dùng nhận được một kết quả cụ thể) và kiểm tra một mẫu trong số đó.
3. *Dựa trên trình duyệt*: Nơi mọi người trong nhóm có thể tìm thấy các tương tác như với công cụ, nhưng sau đó trải nghiệm sự tương tác như người dùng sẽ có, xem những trải nghiệm mà người dùng đã thấy, các nút mà họ đã nhấp vào, kết quả họ nhận được…
4. *Lấy dữ liệu từ người dùng*: Nơi người điều phối có cơ hội gắn cờ các loại tương tác cụ thể và đặt câu hỏi cho người dùng về trải nghiệm của họ. Ví dụ: điều này có thể được thực hiện bằng cách chỉ định một số bối cảnh và sau đó khảo sát một phần nhỏ người dùng tiếp cận bối cảnh đó. Những cuộc khảo sát này có thể yêu cầu người dùng đánh giá trải nghiệm của họ, trả lời một hoặc hai câu hỏi hoặc cho biết họ có kết quả tốt hay xấu.

Tiêu chí đầu tư kiểm tra tương tác:

- Nếu cần một nhóm người rộng rãi để có thể hiểu được các tương tác. Ví dụ: để người tạo trải nghiệm và các bên liên quan trong doanh nghiệp có thể hiểu cách người dùng đang trải nghiệm hệ thống thông minh.
- Nếu cần xây dựng năng lực hỗ trợ để xử lý lỗi. Các công cụ có thể cho phép những người không phải là chuyên gia triển khai tham gia.
- Nếu hệ thống có sai sót gây tốn kém và mong đợi phần lớn việc điều phối sẽ liên quan đến việc giảm thiểu thường xuyên thì đầu tư vào việc kiểm tra các tương tác.

6.2.3. Cân bằng trải nghiệm

Khi có vấn đề, cơ sở người dùng và trí thông minh thay đổi, chúng tạo ra cơ hội để tối ưu hóa trải nghiệm. Ví dụ, khi trí thông minh còn mới và chất

lượng kém thì trải nghiệm có thể rất thụ động. Có lẽ nó không xuất hiện thường xuyên. Có thể nó hiển thị theo cách mà người dùng dễ bỏ qua. Nhưng theo thời gian, trí thông minh sẽ trở nên tốt hơn và người dùng có thể có được kết quả tốt hơn với những trải nghiệm mạnh mẽ hơn. Mức đầu tư để cân bằng trải nghiệm bao gồm:

1. *Đặc biệt*: Nơi thực hiện các thay đổi về trải nghiệm bằng cách thay đổi mã và triển khai lại phần mềm.
2. *Cập nhật tham số*: Nơi người điều phối có thể thay đổi các tham số ảnh hưởng đến trải nghiệm và đẩy các tham số này ra ngoài tương đối rẻ (có thể do trí thông minh được cập nhật). Các thông số có thể gồm (i) Tần suất tương tác; (ii) Màu sắc hoặc kích thước của lời nhắc; (iii) Văn bản sử dụng trong trải nghiệm; (iv) Ngưỡng để tự động hóa một hành động;
3. • *Các lựa chọn thay thế trải nghiệm*: Nơi nhiều trải nghiệm được tạo ra và người điều phối có khả năng chuyển đổi giữa chúng (sử dụng thứ gì đó tương tự như một tham số). Ví dụ, có thể tạo một phiên bản trải nghiệm có lời nhắc tinh tế, một phiên bản có lời nhắc nổi bật và một phiên bản trải nghiệm tự động hóa một hành động. Sau đó, người điều phối có thể chuyển đổi giữa những điều này để đạt được mục tiêu theo thời gian.
4. *Ngôn ngữ trải nghiệm*: Nơi người điều phối có thể soạn thảo và triển khai trải nghiệm ngoài phạm vi thay đổi mã, tương tự như việc tách thông tin thông minh khỏi quá trình triển khai. Điều này có thể được thực hiện bằng cách chỉ định trải nghiệm trong các tập lệnh mà khách hàng tải xuống và thực thi. Hoặc nó có thể được rút ngắn hơn để những người không phải là kỹ sư có thể thực hiện thay đổi một cách an toàn; ví dụ: ngôn ngữ đánh dấu trải nghiệm với thời gian chạy bị hạn chế.

Tiêu chí đầu tư cân bằng trải nghiệm:

- Nếu nhóm điều phối không có tài nguyên kỹ thuật thì có thể tạo một số biện pháp kiểm soát cho trải nghiệm không dựa trên kỹ thuật.
- Nếu phải mất nhiều thời gian để triển khai mã máy khách mới cho khách hàng của mình, có thể đầu tư vào các cách để kiểm soát trải nghiệm thông qua các tham số và cập nhật tệp dữ liệu.

- Nếu cho rằng mình sẽ thay đổi trải nghiệm của mình nhiều lần trong vòng đời của hệ thống thông minh, có thể chọn đầu tư vào việc làm cho việc này trở nên dễ dàng.
- Nếu không điều nào trong số đó là đúng (có tài nguyên kỹ thuật, có thể dễ dàng triển khai mã và không mong đợi quá nhiều thay đổi) thì gần như chắc chắn sẽ rẻ hơn nếu áp dụng cách tiếp cận đặc biệt để trải nghiệm các bản cập nhật.

6.2.4. Ghi đè thông minh

Thông tin quan trọng hơn liên quan đến việc xác định một số bối cảnh và mã hóa thủ công câu trả lời mà thông tin thông minh sẽ cung cấp cho các bối cảnh đó. Có thể làm điều này để sửa chữa những sai lầm tốn kém. Có thể làm điều này để tối ưu hóa một số bối cảnh phổ biến. Có thể làm điều này để hỗ trợ một số mục tiêu kinh doanh (như quảng cáo một phần nội dung trong hệ thống). Có thể không muốn lấn át trí thông minh quá nhiều, nhưng khi cần, sẽ thực sự cần làm vậy.

Mức độ đầu tư cho trí tuệ vượt trội bao gồm:

1. *Đặc biệt*: Nơi dựa vào những người tạo trí thông minh để hack quá trình học tập nhằm cố gắng đạt được kết quả cụ thể (rất khó) hoặc dựa vào các kỹ sư để mã hóa cứng các phần ghi đè mong muốn thành mã và triển khai chúng tới khách hàng.
2. *Là nguồn cấp dữ liệu thông minh*: Trong đó phần ghi đè được coi là nguồn thông tin rất đơn giản được triển khai cùng với thông tin khác của hệ thống và có mức độ ưu tiên cao nhất trong thời gian chạy. Có lẽ được trình bày bằng cách sử dụng tệp dữ liệu ở một số định dạng đơn giản để ánh xạ bối cảnh tới kết quả.
3. *Dựa trên công cụ*: Coi phần ghi đè là nguồn cấp dữ liệu thông minh nhưng tạo ra công cụ để hỗ trợ người điều phối. Các công cụ nên bao gồm các chức năng như sau:
    - Đảm bảo ngữ cảnh cần ghi đè được chỉ định chính xác.
    - Đưa ra phản hồi về mức độ phổ biến của các bối cảnh cụ thể và các phản hồi thông tin hiện tại cho các bối cảnh đó.

- Đưa ra phản hồi về các nội dung ghi đè hiện có, bao gồm số lượng nội dung ghi đè và số lượng người dùng mà chúng tác động.
- Theo dõi xem ai đang thực hiện ghi đè và mức độ tốt/xấu của họ.
- Quản lý hết hạn ghi đè.

4. *Dựa trên trình duyệt*: Nơi các công cụ được kết nối vào bộ hỗ trợ, bao gồm các cách tìm tương tác, xem chúng và ghi đè tất cả chúng ở một nơi.

Tiêu chí đầu tư vào trí tuệ vượt trội:

- Nếu hệ thống có thể phạm phải những sai lầm rất tốn kém, gần như chắc chắn muốn đầu tư vào thông tin quan trọng hơn, có thể bổ sung đặc trưng theo dõi và quản lý luồng công việc xung quanh các công cụ được thảo luận ở đây.
- Thông tin ghi đè cũng có thể là vùng đệm rất hữu ích cho những người sáng tạo thông tin. Điều cuối cùng muốn là nhiều người tham gia hệ thống thông minh sử dụng sản phẩm, tìm ra vô số vấn đề nhỏ và ghi chúng dưới dạng lỗi trong quá trình tạo thông tin.. Có khả năng vượt qua một số vấn đề quan trọng hơn có thể khiến mọi người cảm thấy tốt hơn.

6.2.5. Tạo trí thông minh

Một phần quan trọng của việc điều phối là kiểm soát và tạo ra trí thông minh điều khiển hệ thống thông minh. Sau đây tóm tắt một số khoản đầu tư có thể giúp trí tuệ phát triển và tạo ra tác động.

Đầu tư vào việc tạo ra trí thông minh có thể bao gồm:

1. *Quản lý thông minh*: Bao gồm những điều sau đây:
   - Kiểm soát thời điểm và cách thức cập nhật và triển khai thông tin thông minh.
   - Kiểm soát trí thông minh sống ở đâu.
   - Bổ sung các nguồn thông tin mới vào hệ thống.
   - Kiểm soát cách kết hợp trí thông minh.
2. *Triển khai học máy tự động*: Chúng tạo ra thông tin mới một cách thường xuyên mà có rất ít hoặc không có sự giám sát.

Các hệ thống này có thể cung cấp các biện pháp kiểm soát để thay đổi cách thức tạo ra thông tin thông minh, bao gồm:

- Lượng dữ liệu cần sử dụng.
- Dữ liệu nào sẽ được sử dụng.
- Tần suất chạy bộ.
- Mất bao nhiêu thời gian để tìm kiếm những mô hình tốt.
- Bất kỳ thông số nào khác của quá trình lập mô hình có thể điều chỉnh được.

3. *Môi trường sáng tạo thông minh*: Nơi mà phép đo từ xa nắm bắt bối cảnh và kết quả được thu thập và cung cấp cho học máy, bao gồm thử các thuật toán học máy mới, kỹ thuật đặc trưng mới và kết hợp những hiểu biết mới. Những điều này có thể tạo ra trí thông minh mới đặc biệt hoặc tìm cách cải thiện các hệ thống học tập tự động hiện có.
4. *Hỗ trợ kỹ thuật đặc trưng*: Nơi những người sáng tạo trí tuệ có thể thử các đặc trưng mới rất dễ dàng và nếu họ tìm thấy những đặc trưng hoạt động được thì có thể triển khai chúng cho khách hàng một cách nhanh chóng và an toàn.
5. *Hệ thống đo từ xa*: Nơi người điều phối có thể chỉ định dữ liệu nào cần thu thập và số lượng dữ liệu cần thu thập.

Tiêu chí đầu tư tạo dựng trí tuệ:

Việc tạo ra trí thông minh là một phần quan trọng đối với hệ thống thông minh. Có thể sẽ phải đầu tư vào nhiều hệ thống để hỗ trợ quá trình này.

Các công cụ hiệu quả nhất có xu hướng:

- Giúp ngăn ngừa lỗi.
- Tự động hóa các công việc trần tục.
- Đơn giản hóa các quy trình gồm nhiều bước.
- Để lại một số dấu vết kiểm tra đơn giản.

Khi xây dựng các công cụ hỗ trợ việc tạo ra trí thông minh, hãy tránh gây rối với các công cụ tạo ra trí thông minh cốt lõi (như hệ thống học máy hoặc thời gian chạy). Đây thường là những tiêu chuẩn và việc đổi mới với chúng dường như không phải là cốt lõi đối với đề xuất giá trị.

6.2.6. Tóm tắt

Đã xét nhiều công cụ quan trọng để vận hành hệ thống thông minh. Tiếp sau đây là tóm tắt một số vấn đề chính, bao gồm một số tiêu chí và phương pháp đầu tư vào chúng. Bao gồm các:

- Giám sát các tiêu chí thành công
- Kiểm tra tương tác
- Cân bằng trải nghiệm
- thông minh vượt trội
- Tạo ra trí thông minh

Có thể xây dựng hệ thống thông minh thành công mà không cần đầu tư nhiều vào các lĩnh vực này trong quá trình triển khai. Nhưng có thể sẽ cần thực hiện tất cả các hoạt động này trong vòng đời của hệ thống. có thể quyết định mức độ sử dụng lao động mà muốn và mức độ muốn đầu tư vào công cụ.

Và hãy nhớ rằng việc có các công cụ tốt cho phép thay đổi nhân sự theo thời gian từ những người tạo ra hệ thống (những người thường thích làm việc trên những điều thú vị mới) đến những người điều phối lành nghề (những người cần phải là những người tổng hợp có động lực để đạt được sự xuất sắc bền vững). Đây là những bộ kỹ năng rất khác nhau.

Thông thường, môi trường điều phối sẽ phát triển giống như hệ thống thông minh, với các khoản đầu tư nhỏ được thực hiện để mang lại giá trị cao nhất trong một khoảng thời gian dài cho đến khi không còn khoản đầu tư nào có ý nghĩa nữa.

6.3. Xử lý sai lầm

Sẽ có những sai lầm. Con người tạo ra chúng. Trí tuệ nhân tạo cũng tạo ra chúng. Những sai lầm có thể gây khó chịu hoặc chúng có thể là thảm họa. Mọi hệ thống thông minh nên có chiến lược xác định lỗi; ví dụ: bằng cách theo dõi các số liệu quan trọng và cung cấp cho người dùng những cách dễ dàng để báo cáo sự cố.

Mỗi hệ thống thông minh cũng nên có chiến lược xử lý sai sót. Nó được thực hiện bằng cách cập nhật thông tin thông minh; có lẽ bằng cách yêu cầu con người ghi đè một số hành vi nhất định thủ công; có lẽ bằng cách đưa ra giải pháp thay thế hoặc hoàn lại tiền cho những người dùng bị ảnh

hưởng. Một số sai lầm sẽ rất khó tìm thấy. Một số sẽ rất khó sửa (nếu không đưa ra những lỗi mới). Và một số sẽ mất rất nhiều thời gian để khắc phục (hàng giờ để triển khai một mô hình mới hoặc nhiều tháng để đưa ra chiến lược thông minh mới). Phần này sẽ thảo luận về cách xử lý sai sót, bao gồm các chủ đề sau:

1. Các loại lỗi mà hệ thống có thể mắc phải (đặc biệt là những lỗi xấu).
2. Lý do trí thông minh có thể mắc sai lầm.
3. Cách giảm thiểu sai sót.

Mọi người điều phối hệ thống thông minh nên chấp nhận thực tế về sai sót.

6.3.1. Điều tồi tệ nhất có thể xảy ra

Hãy tự hỏi: Điều tồi tệ nhất mà hệ thống thông minh của tôi có thể làm là gì?

- Có thể hệ thống thông minh sẽ mắc những lỗi nhỏ, chẳng hạn như nháy đèn mà người dùng không quan tâm hoặc phát một bài hát mà họ không yêu thích.
- Có thể nó lãng phí thời gian và công sức, tự động hóa điều gì đó mà người dùng phải hoàn tác hoặc khiến người dùng không chú ý đến điều họ thực sự quan tâm và nhìn vào điều mà trí thông minh đang mắc sai lầm.

- Có thể việc này sẽ khiến doanh nghiệp tốn kém do quyết định tiêu tốn nhiều CPU hoặc băng thông hoặc do vô tình che giấu nội dung hay nhất (và có lợi nhất).

Tìm điều tồi tệ nhất mà hệ thống có thể làm. Sau đó tìm điều tồi tệ thứ hai., thứ ba… Sau đó mời 5 người khác làm điều tương tự. Nắm bắt ý tưởng của họ và chấp nhận chúng.

Và sau đó, khi có mười lăm điều thực sự tồi tệ mà hệ thống thông minh có thể làm, hãy tự hỏi: điều đó có ổn không? Bởi vì những sai lầm kiểu này sẽ xảy ra, chúng sẽ khó phát hiện và khó sửa chữa.

Nếu điều tồi tệ nhất mà hệ thống có thể làm quá tệ đến mức không thể lường trước được, thì có thể muốn thiết kế một hệ thống khác - một hệ thống

không bao giờ có thể làm điều tồi tệ đó, bất kể thông minh có nói gì. Có thể chắc chắn rằng con người là một phần của quá trình đưa ra quyết định. Có thể sử dụng một trải nghiệm ít mạnh mẽ hơn. Có thể sẽ tìm thấy điều gì đó hoàn toàn khác trong cuộc sống của mình…

6.3.2. Những cách phá vỡ tri thức

Một hệ thống thông minh sẽ mắc lỗi vì nhiều lý do khác nhau. Một số trong đó là các vấn đề về triển khai hoặc quản lý; một số trong số đó là vấn đề về trí thông minh. Dưới đây thảo luận về các nguồn vấn đề tiềm ẩn này, bao gồm:

1. Hệ thống ngừng hoạt động
2. Sự cố ngừng hoạt động của mô hình
3. Lỗi thông minh
4. Suy thoái trí tuệ

Bước đầu tiên để sửa lỗi là hiểu nguyên nhân gây ra lỗi đó.

6.3.2.1. Mất hệ thống

Đôi khi máy tính gặp sự cố. Đôi khi Internet chậm. Đôi khi cáp mạng bị cắt. Đôi khi một hệ thống có những lỗi nhỏ trong tương tác hệ thống. Đây là những vấn đề trong quá trình triển khai hoặc vận hành hệ thống thông minh, nhưng chúng có thể biểu hiện giống như các lỗi thông minh xảy ra trong các báo cáo của người dùng, với độ đo lên, xuống.

Việc tách biệt các loại vấn đề này có thể khó khăn trong các hệ thống lớn, đặc biệt khi thông tin được phân tán giữa các máy khách (đang ở các trạng thái nâng cấp khác nhau) và nhiều máy chủ (có thể nằm trong nhiều trung tâm dữ liệu khác nhau). Những sự cố mất điện thảm khốc thường dễ dàng được phát hiện. Nhưng mất điện một phần có thể tinh tế hơn. Ví dụ: giả sử 1% lưu lượng truy cập đến một máy chủ cụ thể và máy chủ đó hoạt động một cách điên cuồng. Một phần trăm người dùng đang gặp phải trải nghiệm tồi tệ và có thể họ đang báo cáo điều đó nhiều lần… Nhưng đó chỉ là 1% cơ sở người dùng. 99% người dùng không bị làm phiền. có bao giờ để ý không? Sự cố ngừng hoạt động của hệ thống hiếm khi xảy ra và chúng cần được khắc phục ngay lập tức. Nếu chúng trở nên phổ biến, chúng sẽ làm tê liệt công tác thông minh; và chúng sẽ gây ảnh hưởng xấu đến tinh thần.

6.3.2.2. Mô hình ngừng hoạt động

Liên quan đến sự cố ngừng hoạt động của hệ thống, sự cố ngừng hoạt động của mô hình là vấn đề triển khai hơn là vấn đề thông tin; nhưng nó sẽ có các triệu chứng tương tự.

Sự cố ngừng hoạt động của mô hình có thể xảy ra khi:

- Tệp mô hình bị hỏng trong quá trình triển khai.
- Tệp mô hình không đồng bộ với mã biến ngữ cảnh thành đặc trưng.
- Môi trường tạo thông tin không đồng bộ với môi trường thời gian chạy thông tin.
- Trí thông minh không đồng bộ với trải nghiệm.

Những vấn đề này có thể rất khó tìm ra. Hãy tưởng tượng nếu một số mã đặc trưng được cập nhật trong môi trường tạo thông tin chứ không phải trong thời gian chạy thông tin. Sau đó, khi một mô hình mới (sử dụng mã đặc trưng được cập nhật) được đẩy vào thời gian chạy (sử dụng mã đặc trưng lỗi thời) sẽ bị nhầm lẫn. Nó sẽ nhận được những giá trị ăn uống mà nó không mong đợi. Nó sẽ phạm sai lầm. Vì lý do này, có thể độ chính xác trong thời gian chạy kém hơn 5% so với trong phòng thí nghiệm. Tất cả các thử nghiệm trong phòng thí nghiệm cho thấy trí thông minh đang hoạt động tốt, nhưng người dùng đang có trải nghiệm kém hơn một chút.

Vì những vấn đề này rất khó tìm ra nên mọi hoạt động triển khai trí thông minh đều phải có các bước kiểm tra và kiểm tra kỹ để đảm bảo môi trường tạo thông tin đồng bộ với môi trường thời gian chạy, mọi thứ đều được triển khai chính xác và tất cả các thành phần đều được đồng bộ hóa.

6.3.2.3. Lỗi thông minh

Khi các mô hình tạo nên trí thông minh không hoàn toàn phù hợp với thế giới (và chúng cũng không phù hợp), sẽ có sai sót. Hãy nhớ lại rằng việc tạo ra trí thông minh là một hành động cân bằng giữa việc học một mô hình rất phức tạp có thể biểu diễn vấn đề và việc học một mô hình có thể khái quát hóa tốt cho các bối cảnh mới. Sẽ luôn có những khoảng trống; những chỗ mà mô hình không hoàn toàn đúng.

Và những lỗ hổng này gây ra những sai lầm, những sai lầm khó có thể sửa chữa thông qua việc sáng tạo trí tuệ. có thể thử một loại mô hình khác,

nhưng nó sẽ mắc phải những loại lỗi (mới) riêng. Có thể nhận được nhiều dữ liệu hơn, nhưng điều đó làm lợi nhuận giảm dần. Có thể thử thêm kỹ thuật đặc trưng, và nó thường hữu ích. Nhưng những sai lầm kiểu này sẽ luôn tồn tại.

Chúng sẽ xuất hiện một chút ngẫu nhiên. Chúng sẽ thay đổi theo thời gian (khi dữ liệu huấn luyện thay đổi). Chúng không dễ sửa; nó đòi hỏi nỗ lực bền bỉ và càng tiến sâu thì càng khó hơn.

Một thách thức nữa đối với các lỗi thông minh là tìm ra phần nào của thông minh chịu trách nhiệm. Khi các mô hình được ghép nối lỏng lẻo (ví dụ: khi chúng có thứ tự thực hiện và mô hình đầu tiên đưa ra tuyên bố về bối cảnh sẽ thắng), có thể dễ dàng xác định chính xác mô hình nào đưa ra câu trả lời sai. Nhưng khi các mô hình được liên kết chặt chẽ (ví dụ: khi đầu ra của một số mô hình được kết hợp bằng cách sử dụng phương pháp phỏng đoán phức tạp hoặc siêu mô hình), lỗi sẽ khó theo dõi hơn. Nếu có sáu mô hình, mỗi mô hình chịu trách nhiệm một phần về lỗi, sẽ bắt đầu từ đâu?

6.3.2.4. Suy thoái trí tuệ

Khi một vấn đề có kết thúc mở, thay đổi theo thời gian thay đổi, trí thông minh mà có ngày hôm qua sẽ không còn tốt như ngày hôm nay. Việc thay đổi vấn đề sẽ tạo ra những lỗi thông minh chung vì những lỗi mới sẽ xảy ra ngay cả khi không thay đổi bất cứ điều gì. Hơn nữa, dữ liệu huấn luyện cho vấn đề *mới* sẽ mất thời gian để tích lũy, nghĩa là có thể phải đợi để phản hồi và có thể không bao giờ có đủ dữ liệu huấn luyện để tìm hiểu kỹ bất kỳ phiên bản cụ thể nào của vấn đề (vào thời điểm tìm hiểu nó, nó không liên quan gì cả).

Có hai loại thay đổi chính:

1. Khi các bối cảnh mới xuất hiện theo thời gian (hoặc các bối cảnh cũ biến mất), trong trường hợp đó, sẽ cần tạo ra thông tin mới để làm việc trên các bối cảnh mới, nhưng dữ liệu huấn luyện hiện có vẫn có thể được sử dụng trên các bối cảnh cũ.
2. Khi ý nghĩa của ngữ cảnh thay đổi theo thời gian, trong trường hợp đó, dữ liệu huấn luyện hiện có có thể gây hiểu nhầm và sẽ cần tập trung vào phép đo từ xa mới để tạo ra thông tin thông minh hiệu quả.

Một cách để hiểu sự xuống cấp trong hệ thống thông minh là bảo tồn các phiên bản cũ của trí thông minh và chạy một loạt các trí thông minh trước đó trên dữ liệu hiện tại, tức trí thông minh của ngày hôm qua, của năm ngày trước, của mười ngày trước…

Bằng cách xem xét những sai lầm thay đổi như thế nào, có thể có được trực giác về cách mà vấn đề đang phát triển và sử dụng trực giác đó khi chọn cách thích ứng.

6.3.3. Giảm nhẹ sai lầm

Những sai lầm ngẫu nhiên, ít tốn kém là điều có thể đoán trước được. Nhưng khi sai lầm tăng đột biến, khi chúng trở nên có hệ thống hoặc khi chúng trở nên rủi ro hoặc tốn kém, có thể cân nhắc các biện pháp giảm thiểu.

Sau đây thảo luận về các phương pháp để giảm thiểu sai sót:

1. Đầu tư vào trí tuệ
2. Cân bằng trải nghiệm
3. Điều chỉnh các thông số quản lý thông minh
4. Sử dụng rào chắn
5. Ghi đè lỗi

6.3.3.1. Đầu tư vào trí tuệ

Trong một hệ thống thông minh lành mạnh, trí thông minh sẽ không ngừng được cải thiện. Một cách để giải quyết sai lầm là chờ đợi trí tuệ bắt kịp. Trên thực tế, hầu hết mọi cách tiếp cận khác để giảm thiểu sai sót đều làm giảm giá trị của hệ thống thông minh đối với những người dùng không gặp sự cố (bằng cách giảm bớt trải nghiệm); hoặc nó làm tăng thêm độ phức tạp và chi phí bảo trì về lâu dài (bằng cách thêm các chỉnh sửa thủ công cần phải duy trì). Vì lý do này, việc nâng cao trí thông minh là một cách tuyệt vời để giải quyết sai lầm khi có thể. Những cách tốt nhất để đầu tư vào việc cải thiện trí thông minh đối với những sai lầm là:

1. Nhận dữ liệu huấn luyện hoặc đo từ xa phù hợp hơn có chứa bối cảnh xảy ra lỗi. Điều này có thể cho phép việc tạo ra trí thông minh bắt đầu giải quyết vấn đề với rất ít công sức.
2. Giúp những người sáng tạo thông tin ưu tiên các phần của hệ thống mà họ dành thời gian bằng cách phân loại lỗi thành các danh mục (theo người dân địa phương, độ tuổi, thuộc tính người

dùng…) và ưu tiên các danh mục. Sau đó, người tạo thông tin có thể làm việc trên các đặc trưng và mô hình hóa giúp ích trong các lĩnh vực cụ thể đó, có thể thông qua phân vùng và mô hình hóa tập trung.

3. Cung cấp thêm nguồn lực cho hoạt động sáng tạo trí tuệ về con người và công cụ.

Những đầu tư này sẽ cải thiện chất lượng tổng thể của thông tin thông minh và theo thời gian. Và có lẽ cách tồi tệ nhất để đầu tư vào trí thông minh là theo dõi các lỗi thông minh như thể chúng là lỗi phần mềm và yêu cầu những người sáng tạo trí thông minh phải chịu trách nhiệm sửa chúng theo thứ tự, từng lỗi một cho đến khi không còn lỗi nào nữa. Đó không phải là cách nó hoạt động. Nếu có lỗi mà nhất định phải sửa thì nên xem xét một trong các phương pháp giảm nhẹ khác.

6.3.3.2. Cân bằng trải nghiệm

Nếu lỗi ở mức độ thấp, lỗi trí tuệ ngẫu nhiên hoặc do trí tuệ suy giảm sẽ khó giải quyết. Trong những trường hợp này, có thể muốn cân bằng lại trải nghiệm, làm cho trải nghiệm bớt mạnh mẽ hơn và ít gây ra lỗi hơn.

Nếu vấn đề đủ nghiêm trọng, về cơ bản có thể cân nhắc việc tắt trải nghiệm thông minh cho đến khi có thể giải quyết được vấn đề. Có nhiều tài liệu thảo luận về cách cân bằng giữa trí thông minh và kinh nghiệm.

6.3.3.3. Điều chỉnh thông số quản lý thông minh

Nếu lỗi là do xuống cấp, tức là do bối cảnh mới xuất hiện nhanh chóng hoặc bối cảnh cũ đang thay đổi ý nghĩa, có thể giải quyết chúng bằng cách huấn luyện và triển khai các mô hình mới nhanh hơn.

Cũng có thể thay đổi dữ liệu huấn luyện mà sử dụng, chẳng hạn như bằng cách loại bỏ dần dữ liệu huấn luyện cũ nhanh hơn (khi ngữ cảnh đang thay đổi ý nghĩa) hoặc lấy mẫu lại các ngữ cảnh mới trong đo từ xa hoặc huấn luyện (giúp thay đổi ý nghĩa và khi ngữ cảnh mới xuất hiện nhanh chóng).

Những cách tiếp cận này tương tự như đầu tư vào trí thông minh, nhưng chúng có thể phản ứng nhanh hơn. Ví dụ: khi kỳ nghỉ lễ sắp đến, một sự kiện thời tiết xấu xảy ra hoặc một sản phẩm mới được tung ra thị trường, vấn đề thay đổi nhanh hơn bình thường. Người điều phối có thể biết điều này

và điều chỉnh một số nút thay vì chờ đợi những người tạo ra thông tin học cách dự đoán những loại sự kiện này.

6.3.3.4. Sử dụng rào chắn

Đôi khi gặp phải những lỗi lầm ngớ ngẩn. Bất kỳ con người nào cũng sẽ nhìn vào sai lầm và biết rằng nó không thể đúng được. Ví dụ:

- Khi lò nướng viên ở nhiệt độ 800 độ, không bao giờ muốn đổ thêm dầu vào lửa.
- Với trẻ 10 tuổi, không bao giờ cho chúng xem phim kinh dị.

Trong những trường hợp này, có thể thử đánh lừa các thuật toán tạo trí thông minh để tìm hiểu những điều này. Có thể thu thập dữ liệu huấn luyện tập trung. Có thể săn lùng những người sáng tạo ra thông tin thông minh. Có thể đầu tư nhiều tháng làm việc…

Hoặc có thể triển khai một phương pháp phỏng đoán đơn giản để ghi đè trí thông minh khi nó chuẩn bị làm điều gì đó rõ ràng là sai: rào chắn bảo vệ.

Khi sử dụng rào chắn, hãy đảm bảo:

- Hãy thận trọng, chỉ ghi đè lên những vấn đề rõ ràng và đừng bị lôi kéo vào việc tạo ra những thông tin phức tạp thủ công.
- Xem lại các quyết định, bằng cách theo dõi hiệu suất (và chi phí) của các rào cản và loại bỏ hoặc nới lỏng những rào cản trở nên ít quan trọng hơn khi trí thông minh được cải thiện hoặc vấn đề thay đổi.

6.3.3.5. Ghi đè lỗi

Đôi khi không có cách nào khác; hệ thống sẽ mắc phải những sai lầm đắt giá mà không thể giảm thiểu bằng bất kỳ cách nào khác và sẽ phải khắc phục những sai lầm này thủ công. Ví dụ:

- Điều hành một dịch vụ chống thư rác và dịch vụ này đang xóa tất cả thư từ một doanh nghiệp hợp pháp.
- Chạy một công cụ tìm kiếm trang web hài hước và nó đánh dấu trang web trò đùa phổ biến nhất trên Internet là không hài hước.

Khi những lỗi này đủ quan trọng, có thể muốn có trải nghiệm người dùng đặc biệt để cho phép người dùng báo cáo sự cố. Và có thể có một số

quy trình liên quan đến việc phản hồi các báo cáo này. Ví dụ: có thể tạo một nhóm hỗ trợ với các công cụ và quy trình làm việc phù hợp để kiểm tra mọi lỗi được báo cáo trong vòng một giờ, 24 giờ một ngày, 7 ngày một tuần.

Giống như các rào chắn, hãy đảm bảo sử dụng biện pháp ghi đè một cách tiết kiệm và theo dõi chất lượng cũng như chi phí của việc ghi đè theo thời gian.

6.3.4. Tóm tắt

Sai lầm là một phần của hệ thống thông minh và nên có kế hoạch đo lường sai sót và xử lý chúng. Một phần của kế hoạch này là hiểu những điều xấu mà hệ thống có thể gây ra. Hãy thành thật với chính mình. Hãy sáng tạo trong việc tưởng tượng các vấn đề. Để khắc phục sự cố, việc biết nguyên nhân gây ra sự cố là rất hữu ích. Những sai sót có thể xảy ra khi:

1. Một phần của hệ thống thông minh bị ngừng hoạt động.
2. Mô hình được tạo, triển khai hoặc diễn giải không chính xác.
3. Trí thông minh không phải là giải pháp hoàn hảo để giải quyết vấn đề (và thực tế không phải vậy).
4. Sự cố hoặc thay đổi cơ sở người dùng.

Sau khi phát hiện ra vấn đề, có thể giảm thiểu vấn đề đó bằng một số cách:

- Bằng cách đầu tư nhiều hơn vào trí tuệ.
- Bằng cách cân bằng lại trải nghiệm.
- Bằng cách thay đổi các thông số quản lý thông minh.
- Bằng cách thực hiện các rào chắn.
- Bằng cách ghi đè lỗi.

Kế hoạch giảm thiểu sai sót chủ động có thể cho phép phần còn lại của hệ thống thông minh hoạt động tốt hơn, và đạt được nhiều tác động hơn. Chấp nhận những sai lầm, khôn ngoan và hiệu quả trong việc giảm thiểu chúng là một phần quan trọng trong việc điều phối một hệ thống thông minh.

6.4. Đối thủ và lạm dụng

Bất cứ khi nào tạo ra thứ gì đó có giá trị, sẽ có người cố gắng kiếm tiền từ nó. Hệ thống thông minh cũng không khác. Nếu dành năng lượng, tiền bạc và thời gian để thu hút người dùng, ai đó sẽ cố gắng kiếm tiền từ những người dùng đó. Nếu xây dựng một doanh nghiệp đang gây áp lực lên đối thủ

cạnh tranh, ai đó sẽ cố gắng gây khó khăn hơn cho trong việc điều hành doanh nghiệp đó.

Sau đây là một số cách phổ biến mà việc lạm dụng có thể ảnh hưởng đến hệ thống thông minh:

- Những kẻ lạm dụng cố gắng kiếm tiền từ người dùng, ví dụ như bằng cách gửi thư rác cho họ.
- Những kẻ lạm dụng cố gắng đánh cắp thông tin về hệ thống hoặc người dùng để sao chép hoặc bán.
- Những kẻ lạm dụng cố gắng sử dụng nền tảng để tổ chức các cuộc tấn công vào các hệ thống khác.
- Những kẻ lạm dụng cố gắng đầu độc hệ thống để nó không hoạt động theo cách mong muốn.

Một số hoạt động này là bất hợp pháp, nhưng một số thì không. Và ngay cả khi các hoạt động này là bất hợp pháp, bản chất toàn cầu của Internet khiến việc tìm ra những kẻ tấn công trở nên rất khó khăn và càng khó truy tố chúng hơn.

Vì điều này, tất cả các hệ thống thông minh thành công cần phải được chuẩn bị để tự bảo vệ mình khỏi bị lạm dụng.

Phần này đề cập các khái niệm cơ bản về lạm dụng để có thể hiểu được thách thức, sẵn sàng xác định hành vi lạm dụng khi nó xảy ra với và có một số công cụ giúp hệ thống thông minh khó bị lạm dụng hơn.

6.4.1. Lạm dụng là một công việc kinh doanh

Điều đầu tiên cần biết về lạm dụng là nó là một công việc kinh doanh. Đại đa số những người thực hiện hành vi lạm dụng đều làm việc đó để kiếm tiền (mặc dù một số hành vi lạm dụng được thực hiện để giải trí, vì công bằng xã hội hoặc để hỗ trợ hoạt động thông minh). Một số cách mà những kẻ lạm dụng có thể kiếm tiền bao gồm:

- *Chuyển truy cập*: Bằng cách đánh lừa hệ thống thông minh để hiển thị cho người dùng những nội dung mà kẻ lạm dụng muốn họ xem. Đây thực chất là quảng cáo. Kẻ lạm dụng có được thỏa thuận với một trang web và trả tiền người dùng để họ hướng từ trang web của họ đến trang web khác. Điều này thường được gọi là *gửi thư rác*.

- *Xâm phạm thông tin cá nhân*: Bao gồm số an sinh xã hội, thông tin liên lạc, mật khẩu, thông tin ngân hàng… Những kẻ lạm dụng có thể sử dụng thông tin này trực tiếp hoặc bán lại nó để kiếm lợi nhuận nhanh chóng cho những kẻ lạm dụng khác.
- *Xâm nhập máy tính*: Bằng cách lừa người dùng cài đặt những thứ xấu trên máy tính của họ. Khi chúng gặp lỗi trên máy tính của người dùng, chúng có thể lấy cắp thông tin cá nhân hoặc sử dụng máy tính của người dùng để tiến hành các cuộc tấn công tiếp theo. Khi những kẻ lạm dụng có thể sử dụng hệ thống thông minh để liên lạc với người dùng, họ có thể lừa họ làm đủ thứ điều điên rồ.
- *Tăng cường nội dung*: Bằng cách đánh lừa hệ thống thông minh hành xử theo cách chúng muốn. Ví dụ: kẻ lạm dụng có thể đưa ra các đánh giá giả mạo về một sản phẩm trên một trang web thương mại điện tử để làm cho sản phẩm đó trở nên hấp dẫn, nổi bật hơn và bán được nhiều hơn.
- *Ngăn chặn nội dung*: Bằng cách cố gắng làm tổn thương hệ thống thông minh hoặc bằng cách cố gắng làm hại những người dùng khác trong hệ thống. Ví dụ: kẻ lạm dụng có thể báo cáo nội dung của đối thủ cạnh tranh là xúc phạm.
- *Trộm cắp trực tiếp*: Nội dung, có thể để bán lại, chẳng hạn như ăn cắp nội dung số trong trò chơi trực tuyến.

Những kẻ lạm dụng đã tạo ra thị trường cho tất cả những thứ này. Vì vậy, rất dễ dàng cho bất kỳ kẻ lạm dụng nào tìm cách thực hiện những việc như thế này trên hệ thống thông minh để biến hoạt động đó thành tiền.

6.4.2. Lạm dụng từng chút một

Hãy tưởng tượng đang tìm một hoạt động mà có thể làm để kiếm được một phần mười xu. Bấm vào một nút, chọn một tùy chọn, bấm vào nút khác. Một phần mười xu xuất hiện trong tài khoản ngân hàng.

Hình. Lạm dụng mạng.

Nghe có vẻ vô nghĩa. sẽ phải thực hiện hoạt động đó hàng nghìn lần chỉ để kiếm được một đô la, một trăm nghìn lần để kiếm được một trăm đô la. Thật là lãng phí thời gian! Nhưng bây giờ hãy tưởng tượng có thể lập trình cho một chiếc máy tính làm việc đó, và chiếc máy tính đó có thể làm việc đó hàng triệu lần một giờ, mỗi giờ, cho đến mãi mãi. Đây là lạm dụng Internet. Nói chung, lạm dụng Internet liên quan đến một hoạt động rất khó thành công (chẳng hạn như lừa ai đó tiết lộ mật khẩu) hoặc có rất ít giá trị mỗi khi thành công (như gửi lưu lượng truy cập đến một trang web); nhưng kẻ lạm dụng thực hiện những điều này ở mức thấp các hoạt động có giá trị hơn. Và họ kiếm tiền tốt khi làm việc đó. Điều này có nghĩa là một ngày nào đó có thể không gặp phải vấn đề lạm dụng. Cho rằng mình ổn, nhưng kẻ lạm dụng có thể đang thử nghiệm, thử các hoạt động khác nhau, đo lường tần suất người dùng rơi vào bẫy lừa đảo của họ hoặc lượng truy cập mà họ có thể tạo ra từ trang. người dùng, thực hiện phép tính; và khi họ tìm thấy phép tính phù hợp với mình, họ có thể mở rộng quy mô một cách nhanh chóng.

Rất dễ để tình trạng lạm dụng từ con số 0 trở thành thảm họa chỉ sau một đêm.

6.4.2.1. Ước tính rủi ro

Hệ thống thông minh sẽ trở nên thú vị đối với những kẻ lạm dụng nếu bất kỳ điều nào sau đây là đúng:

- *Nó có rất nhiều người dùng*: Khi hệ thống thông minh trở nên phổ biến hơn, nó sẽ có nhiều người dùng hơn. Điều này có nghĩa là những kẻ lạm dụng có thể mở rộng quy mô các cuộc tấn công của họ hơn nữa và kiếm được nhiều tiền hơn cho mỗi cuộc tấn công. Thật đáng để họ dành thời gian thử nghiệm hệ thống thông minh

bởi vì nếu họ tìm ra cách kiếm được một phần mười xu, họ có thể kiếm được rất nhiều tiền.

- *Những kẻ lạm dụng có thể sử dụng hệ thống để liên lạc với người dùng*: Đặc biệt nếu họ có thể đưa URL vào thông tin liên lạc của mình. Thông tin liên lạc có thể là một tin nhắn, một email, một trang web, một bức ảnh, một bài đánh giá, một bình luận… Những kẻ lạm dụng sẽ tìm cách kiếm tiền từ việc giao tiếp với người dùng bằng cách lừa họ và gửi thư rác cho họ.
- *Nó tương tác với nội dung do người dùng tạo*: Và trí thông minh đóng bất kỳ vai trò nào trong việc quyết định nội dung nào sẽ hiển thị, cách chú thích hoặc hiển thị nội dung hoặc cách sắp xếp nội dung mà nó hiển thị. Việc tác động đến cách người dùng xem nội dung sẽ ảnh hưởng đến nội dung mà họ tương tác; và mọi khoản tiền liên quan sẽ chảy như thế nào. Những kẻ lạm dụng sẽ cố gắng thâm nhập vào vòng lặp đó.
- *Những sai lầm mà nó gây ra khiến ai đó phải trả giá*: Đặc biệt nếu chi phí đó có thể hướng tới các bên cụ thể. Ví dụ: khi máy nướng bánh mì thông minh đốt cháy một nhãn hiệu bánh tart đông lạnh cụ thể; hoặc khi hệ thống thông minh đang loại bỏ đối thủ cạnh tranh không thông minh.
- *Nó có thể làm bất cứ việc gì khác mà một người tận tâm có thể kiếm tiền*: Hãy coi những kẻ lạm dụng là những hacker thông minh với đạo đức đáng ngờ và có rất nhiều thời gian trong tay.

6.4.2.2. Nhận dạng vấn đề lạm dụng

Khi những kẻ lạm dụng mở rộng quy mô tấn công vào hệ thống thông minh, họ sẽ tạo ra những kiểu sử dụng kỳ quặc. Thường có thể phát hiện ra chúng bằng phương pháp đo từ xa bằng cách tìm kiếm:

- Nhóm lớn người dùng sử dụng hệ thống thông minh theo những cách suy thoái (rất tập trung vào các phần giúp họ kiếm tiền).
- Các bối cảnh cho thấy hoạt động tăng đột biến so với mức sử dụng thông thường.
- Bối cảnh trong đó việc phân phối kết quả thay đổi đáng kể (vì kẻ xấu đang gửi cho thông tin không chính xác).
- Các mẫu khiếu nại và báo cáo vấn đề của người dùng.

Những kẻ lạm dụng có thể cố gắng trà trộn vào, nhưng họ sẽ khó bắt kịp các hoạt động của người dùng hợp pháp, vì vậy thường có thể phát hiện ra các cuộc tấn công của họ nếu dành thời gian tìm kiếm chúng. Cũng có thể thường xuyên phát hiện các cuộc tấn công của họ bằng cách đặt cảnh báo về những thay đổi mạnh mẽ. Lưu lượng truy cập vào một phần hệ thống tăng lên rất nhiều?

6.4.3. Cách chống lạm dụng

Nếu lạm dụng trở thành một vấn đề, một số cách tiếp cận giải quyết bao gồm:

1. Thêm chi phí vào sản phẩm.
2. Trở nên ít thú vị hơn đối với những kẻ lạm dụng.
3. Học máy với đối thủ.
4. Đưa đối thủ ra khỏi vòng kiểm soát.

6.4.3.1. Tăng chi phí

Có thể ngừng lạm dụng và kiếm nhiều tiền hơn! Tính phí nhiều hơn cũng có thể khiến khách hàng hợp pháp lo lắng, vì vậy đây có thể không phải là một lựa chọn. Nhưng hãy nhớ rằng lạm dụng là một công việc kinh doanh và cách tốt nhất để ngăn chặn hành vi lạm dụng là khiến những kẻ lạm dụng khó kiếm lợi hơn.

Ví dụ: việc kiếm lợi từ việc lạm dụng sẽ trở nên khó khăn hơn nếu kẻ lạm dụng cần:

- Trả số tiền nhỏ cho mỗi tài khoản trên hệ thống thông minh.
- Nhập một số ký tự nguệch ngoạc cho mỗi đánh giá mà họ muốn để lại.
- Mua một máy nướng bánh mì thông minh cho mỗi cuộc tấn công mà chúng muốn thực hiện.

Những chi phí này hiệu quả nhất khi chúng tác động đến những kẻ lạm dụng nhiều hơn là tác động đến những người dùng hợp pháp. Ví dụ: nếu mỗi người dùng tốt phải nhập các ký tự nguệch ngoạc một lần, nhưng kẻ ngược đãi cần nhập chúng cho mọi hoạt động họ thực hiện. Việc tăng thêm chi phí có thể khiến những kẻ lạm dụng ngừng hoạt động mà thậm chí những người dùng hợp pháp cũng không biết điều đó đang xảy ra.

6.4.3.2. Trở nên ít thú vị hơn đối với những kẻ lạm dụng

Có thể thay đổi sản phẩm của mình để làm ít điều mà những kẻ lạm dụng thấy thú vị hơn, ví dụ:

- Loại bỏ hoặc hạn chế các kênh liên lạc.
- Lưu trữ ít thông tin cá nhân của người dùng trên trang web và cẩn thận về nơi hiển thị thông tin đó.
- Giảm tác động của phản hồi của người dùng về cách trình bày nội dung.

Những điều này cũng có thể làm cho sản phẩm kém hữu ích hơn đối với người dùng thực, nhưng đôi khi một hoặc hai điều chỉnh nhỏ sẽ tạo ra sự khác biệt trong việc phá vỡ khả năng kiếm lợi của những kẻ lạm dụng.

6.4.3.3. Học máy với đối thủ

Có thể sử dụng công nghệ học máy để xác định các tương tác lạm dụng rồi xóa chúng. Học máy là một công cụ tốt, nhưng những kẻ lạm dụng rất giỏi trong việc thay đổi các cuộc tấn công của họ theo kiểu đánh lừa học máy. Những kẻ lạm dụng sẽ ít thay đổi cuộc tấn công so với việc phải theo đuổi chúng và thay đổi khả năng học máy của mình.

Có thể thực hiện học máy để chống lạm dụng và nó có thể sẽ hữu ích, nhưng nên xem xét các lựa chọn khác trước và đảm bảo rằng hiểu học máy thực sự có thể tác động như thế nào đến mô hình kinh doanh của kẻ lạm dụng trước khi đầu tư quá nhiều.

6.4.3.4. Đưa kẻ lạm dụng ra khỏi cuộc chơi

Bất cứ khi nào xác định được hành vi lạm dụng, nên chặn mọi thứ mà kẻ lạm dụng đã sử dụng để khởi động cuộc tấn công, bao gồm tài khoản, máy nướng bánh mì, máy nướng viên, trang web vui nhộn, hệ thống phun nước… Điều này sẽ đảm bảo những kẻ lạm dụng phải trả giá cao nhất có thể khi chúng mở rộng quy mô các cuộc tấn công. Cơ sở hạ tầng mà họ dùng để tấn công ngày hôm qua đã bị đốt cháy nên hôm nay họ cần phải ra ngoài và xây dựng lại cơ sở hạ tầng.

Một lựa chọn khác là chỉ tập trung vào việc tạo ra trí thông minh từ những người dùng đáng tin cậy. Hãy tưởng tượng có 100.000 người dùng đã đồng hành trong nhiều năm, sử dụng dịch vụ thông minh, tạo ra dữ liệu đo từ xa và bối cảnh với kết quả huấn luyện. Những người dùng này khá an toàn;

họ không phải là tài khoản do những kẻ lạm dụng tạo ra để thực hiện một số cuộc tấn công mới. Bằng cách hạn chế việc tạo thông tin thông minh đối với những người dùng *nổi tiếng tốt*, thường có thể tránh hoàn toàn việc lạm dụng.

6.4.4. Tóm tắt

Bất cứ khi nào tạo ra thứ gì đó có giá trị, những kẻ lạm dụng sẽ đến và cố gắng hưởng lợi từ công việc khó khăn, khiến người dùng và hệ thống thông minh gặp nguy hiểm. Phần lớn hành vi lạm dụng được thực hiện để kiếm tiền. Bằng cách hiểu cách những kẻ lạm dụng kiếm tiền, có thể kiểm soát mức độ thú vị của hệ thống thông minh đối với họ. Thông thường, cách rẻ nhất để chống lạm dụng là thực hiện một vài điều chỉnh nhỏ trong cách hệ thống hoạt động, để những kẻ lạm dụng không thể tìm ra cách kiếm được lợi nhuận đáng tin cậy.

Lạm dụng thường nhắm vào các hoạt động có giá trị thấp và có thể mở rộng quy mô đáng kể; một phần mười xu, một triệu lần mỗi ngày. Thường có thể coi việc lạm dụng trong phép đo từ xa là các hoạt động tăng đột biến không phù hợp với thói quen sử dụng thông thường. Có thể không ngăn chặn được việc này trong thời gian thực, nhưng thường có thể biết liệu mình có đang bị tấn công hay không.

Một số thực hành có thể ngăn cản sự lạm dụng:

- Tăng chi phí lạm dụng.
- Thay đổi hệ thống thông minh để ít có giá trị hơn đối với những kẻ lạm dụng.
- Sử dụng một số công nghệ học máy;
- Tin tưởng những người dùng đã thiết lập nhiều hơn những người dùng mới và xóa tất cả những người dùng có liên quan đến hành vi lạm dụng đã được xác nhận.

6.5. Tiếp cận hệ thống thông minh của riêng

Bây giờ đã có nền tảng để thực hiện dự án hệ thống thông minh của riêng mình khi biết:

1. *Cách tiếp cận hệ thống thông minh*: Nó tốt cho mục đích gì, khi nào cần và cách đặt mục tiêu cho hệ thống.

2. *Cách thiết kế một trải nghiệm thông minh*: Một trải nghiệm giúp đạt được mục tiêu và tạo ra dữ liệu giúp phát triển trí thông minh.
3. *Cần những gì để triển khai hệ thống thông minh*: Cách thực thi, quản lý và đo lường trí thông minh.
4. *Cách tạo ra trí thông minh*: Bao gồm nhiều cách tiếp cận, nhưng đặc biệt là sử dụng học máy.
5. *Cách điều phối một hệ thống thông minh*: Để kết hợp các bộ phận này lại với nhau trong suốt vòng đời của nó và đạt được tác động mà mong muốn.

Tiếp sau đây là các khái niệm chính và cung cấp danh sách kiểm tra để tiếp cận dự án hệ thống thông minh của riêng.

6.5.1. Danh sách kiểm tra hệ thống thông minh

Hệ thống thông minh đang thay đổi thế giới bằng cách khép lại vòng lặp giữa người dùng và trí thông minh. hệ thống thông minh giải quyết các vấn đề quan trọng, làm hài lòng người dùng và giúp các tổ chức đạt được thành công.

Để tạo ra một hệ thống thông minh, cần phải làm nhiều việc và phải đưa ra nhiều quyết định. Sau đây tập hợp những điểm chính trong số này vào một nơi, kèm theo các tham chiếu đến các chương khác để có thể tìm thêm thông tin chi tiết để trợ giúp.

Khi tiếp cận một dự án hệ thống thông minh, nên xem xét các bước sau.

6.5.1.1. Tiếp cận dự án hệ thống thông minh

1. Bắt đầu bằng cách đảm bảo hệ thống thông minh phù hợp.

Hệ thống thông minh rất hữu ích khi mong muốn thay đổi tính thông minh của sản phẩm nhiều lần trong suốt vòng đời của nó. Điều này thường cần thiết khi vấn đề là:

- Lớn.
- Kết thúc mở.
- Thời gian thay đổi.
- Bản chất khó khăn.

Ngoài ra, hệ thống thông minh hoạt động tốt nhất khi:

- Giải pháp từng phần là khả thi và thú vị.
- có thể sử dụng dữ liệu từ người dùng tương tác với hệ thống để cải thiện nó.
- có thể tác động đến một mục tiêu có phạm vi phù hợp và có ý nghĩa.
- Vấn đề biện minh cho nỗ lực xây dựng hệ thống thông minh.

2. Xác định thành công sẽ như thế nào đối với hệ thống thông minh.

Tạo sự đồng thuận về những gì hệ thống thông minh nên thực hiện theo cách truyền đạt kết quả mong muốn, có thể đạt được và có thể đo lường được. Quyết định những mục tiêu tổ chức nào (như lợi nhuận hoặc số lượng sản phẩm đã bán) mà hệ thống thông minh sẽ đóng góp và những chỉ số hàng đầu nào có thể sử dụng để đạt được phản hồi rằng đang đạt được tiến bộ. Sau đó, quyết định hệ thống thông minh nên tối ưu hóa hàng ngày bằng cách xác định kết quả người dùng mà muốn tạo và các thuộc tính mô hình sẽ rất quan trọng để thành công. Đưa ra một kế hoạch để đo lường xem hệ thống thông minh có đang đạt được các mục tiêu hay không, có thể bao gồm: đo từ xa, chờ kết quả trở nên rõ ràng, sử dụng đánh giá của con người và hỏi người dùng.

6.5.1.2. Lập kế hoạch cho trải nghiệm thông minh

3. Quyết định cách trình bày trí thông minh của hệ thống cho người dùng để đạt được mục tiêu.

Mục tiêu của trải nghiệm thông minh là:

- Trình bày trí thông minh cho người dùng.
- Đạt được các mục tiêu của hệ thống.
- Giảm thiểu những sai sót về trí tuệ.
- Tạo dữ liệu giúp nâng cao trí thông minh.

Để đạt được những điều này, cần cân bằng trải nghiệm bằng cách đánh đổi giữa:

- Sức mạnh của trải nghiệm.
- Tần suất tương tác.
- Giá trị của sự thành công.
- Cái giá của sai lầm (cả phát hiện và khắc phục).
- Chất lượng của trí thông minh.

Cân bằng trải nghiệm thông minh là một quá trình trong đó trải nghiệm thay đổi khi trí thông minh thay đổi. Có thể chừa cho mình một khoảng trống để lặp lại. Các phương thức tương tác cụ thể giữa người dùng và trí thông minh mà có thể sử dụng như một phần của sự cân bằng này, bao gồm:

- Tự động hóa
- Nhắc nhở
- Tổ chức
- Chú thích

Gần như chắc chắn sẽ sử dụng một số kết hợp của những thứ này trong hệ thống thông minh của mình để trải nghiệm mạnh mẽ hơn khi thông tin chắc chắn và ít mạnh mẽ hơn khi thông tin không chắc chắn.

4. Lên kế hoạch thu thập dữ liệu từ trải nghiệm.

Trải nghiệm thông minh lý tưởng sẽ tạo ra dữ liệu cho phép cải thiện hệ thống thông minh (và cụ thể là chính trí thông minh) theo thời gian.

Để hữu ích cho việc cải thiện hệ thống thông minh, dữ liệu phải:

- Bao gồm bối cảnh, hành động của người dùng và kết quả của sự tương tác.
- Cung cấp phạm vi bao quát tốt cho các phần của hệ thống thông minh và không gian có vấn đề.
- Thể hiện sự tương tác thực sự với người dùng.
- Hãy khách quan.
- Tránh vòng lặp phản hồi.
- Có đủ quy mô để hữu ích cho việc tạo ra trí thông minh.

Tốt nhất khi dữ liệu này được tạo ngầm định, như một sản phẩm phụ tự nhiên của quá trình người dùng tương tác với hệ thống thông minh. Nhưng đôi khi cần yêu cầu trợ giúp để hiểu cách người dùng nhìn nhận kết quả mà họ nhận được.

Các cách để nhận được phản hồi rõ ràng của người dùng bao gồm:

- Cho phép người dùng đưa ra xếp hạng trong trải nghiệm (chẳng hạn như không thích, không thích hoặc 1-5 sao).
- Chấp nhận báo cáo của người dùng về kết quả xấu trong trải nghiệm (chẳng hạn như nút *báo cáo là thư rác*).

- Tạo tầng hỗ trợ để người dùng có thể báo cáo vấn đề để được trợ giúp (gọi số điện thoại để được trợ giúp).
- Nhắc người dùng phân loại một số kết quả mà họ chọn.

5. Lập kế hoạch để xác minh trải nghiệm thông minh.

Cần lập kế hoạch để:

- Biết rằng trải nghiệm đang hoạt động chính xác (bất chấp mọi sai lầm mà trí thông minh đang mắc phải).
- Biết rằng trải nghiệm đang góp phần giúp hệ thống thông minh đạt được các mục tiêu.

Để giải quyết vấn đề này, có thể lập kế hoạch cho các công cụ để kiểm tra trải nghiệm sẽ được tạo cho bất kỳ bối cảnh cụ thể nào cũng như các công cụ giúp tạo và nắm bắt các bối cảnh nơi xảy ra sự cố.

Cũng có thể muốn lập kế hoạch sử dụng các phiên bản thông tin khác nhau của hệ thống trong các công cụ này, bao gồm cả thông tin trực tiếp; một bức ảnh chụp nhanh về trí thông minh và một số thông tin đơn giản hoặc còn sơ khai.

6.5.1.3. Lập kế hoạch triển khai hệ thống thông minh

Việc triển khai hệ thống thông minh đòi hỏi tất cả công việc triển khai hệ thống truyền thống, cộng với công việc để kích hoạt trí thông minh của hệ thống. Những thành phần bổ sung này bao gồm:

- Thời gian chạy thông minh.
- Quản lý thông minh.
- Đo từ xa giúp phát triển trí thông minh.
- Môi trường sáng tạo trí tuệ.
- Công cụ điều phối thông minh.

Các thành phần này cung cấp các khả năng quan trọng để cải thiện hệ thống thông minh trong suốt vòng đời của nó. Xây dựng cách triển khai phù hợp có thể giúp việc tạo ra thông tin và điều phối hệ thống thông minh dễ dàng hơn nhiều.

6. Quyết định trí thông minh sẽ ở đâu.

Trí thông minh có thể là bất kỳ điều nào sau đây:

- Thông minh tĩnh trong sản phẩm.
- Thông tin phía khách hàng.
- Trí tuệ lấy máy chủ làm trung tâm.
- Thông tin back-end

Xác định vị trí tri thức theo các hiệu ứng sau:

- Độ trễ trong việc cập nhật trí thông minh.
- Độ trễ trong việc thực thi trí thông minh.
- Chi phí vận hành hệ thống thông minh.
- Mọi yêu cầu về hoạt động ngoại tuyến.

7. Thiết kế thời gian chạy thông minh.

Thời gian chạy thông minh là nơi thông tin được thực thi và trải nghiệm được cập nhật. Thời gian chạy thông minh phải thực hiện tất cả những điều sau:

- Thu thập bối cảnh cần để trí thông minh có thể đưa ra quyết định đúng đắn.
- Trích xuất các đặc điểm từ ngữ cảnh theo cách an toàn nhưng hỗ trợ đổi mới trong việc tạo ra thông tin.
- Xử lý các tập tin mô hình và cập nhật chúng.
- Thực thi các mô hình trên ngữ cảnh (và các đặc trưng của chúng).
- Lấy kết quả và dự đoán do quá trình thực thi tạo ra và thắp sáng trải nghiệm thông minh.

8. Kế hoạch quản lý thông minh.

Quản lý thông minh là hệ thống để thêm thông tin mới vào hệ thống và bật nó cho người dùng một cách an toàn. Điều nay bao gôm:

- Đưa trí tuệ vào hệ thống thông minh.
- Kiểm tra thông tin thông minh để đảm bảo rằng nó rõ ràng không có hại hoặc bị hỏng.
- Kết hợp thông tin thông minh khi có nhiều nguồn thông tin thông minh, bao gồm cả việc đảm bảo tất cả chúng đều đồng bộ.
- Thắp sáng trí tuệ cho người dùng một cách có kiểm soát. Điều đó có thể bao gồm thông tin thông minh thầm lặng, triển khai có kiểm soát, điều hành (thử nghiệm A/B) và hỗ trợ đảo ngược.

9. Kế hoạch đo từ xa.

Đo từ xa là một liên kết quan trọng để kết nối việc sử dụng với trí thông minh, do đó hệ thống thông minh có thể cải thiện theo thời gian. Đo từ xa được sử dụng để đảm bảo hệ thống thông minh đang hoạt động, hiểu kết quả mà người dùng nhận được và thu thập dữ liệu để phát triển trí thông minh. Có nhiều thứ trong một hệ thống hơn mức có thể đo lường được, vì vậy cần đưa ra một số quyết định về cách kiểm soát quy mô của những gì thu thập được trong phép đo từ xa. Nhưng cũng cần cung cấp các công cụ để cho phép những người sáng tạo và điều phối trí thông minh thích ứng theo vấn đề và hệ thống thông minh thay đổi theo thời gian. Những công cụ này bao gồm:

- Lấy mẫu những gì được thu thập trong phép đo từ xa và tăng giảm tốc độ lấy mẫu theo thời gian.
- Tóm tắt các tương tác theo cách cho phép thu thập thông tin phù hợp một cách hiệu quả.
- Hỗ trợ nhắm mục tiêu linh hoạt theo bối cảnh và người dùng để lấy mẫu dữ liệu từ xa.

6.5.1.4. Hãy sẵn sàng để tạo ra trí thông minh

Thông minh là một phần của hệ thống thông minh giúp đưa ra các quyết định khó khăn, ánh xạ từ bối cảnh đến dự đoán, hỗ trợ trải nghiệm thông minh, tạo ra giá trị cho người dùng và cho tổ chức.

10. Chuẩn bị đánh giá trí thông minh.

Với trí thông minh, đánh giá là sáng tạo. Có thể dễ dàng đo lường bất kỳ trí thông minh nào để biết:

- Nó khái quát tốt như thế nào.
- Các loại sai lầm mà nó mắc phải.
- Sự phân bố sai lầm của nó.

Để làm được điều này, cần dữ liệu đánh giá, hy vọng là từ dữ liệu đo từ xa. Đây phải là dữ liệu đầy đủ để có được những đánh giá có ý nghĩa thống kê. Nó cũng phải độc lập với dữ liệu huấn luyện (đôi khi phức tạp) và nó sẽ cho phép hiểu trí thông minh hoạt động như thế nào đối với các nhóm nhỏ quan trọng.

11. Quyết định cách tổ chức trí thông minh của mình.

Việc tổ chức sáng tạo thông tin rất quan trọng khi làm việc với hệ thống thông minh lớn, cần:

- Có nhiều người cùng hợp tác tạo ra trí thông minh.
- Dọn dẹp những sai lầm mà trí thông minh đang mắc phải.
- Giải quyết phần dễ một cách dễ dàng và tập trung các kỹ thuật phức tạp hơn vào phần khó của vấn đề.
- Kết hợp thông tin thông minh kế thừa hoặc thông tin thu được từ các nguồn bên ngoài.

Dưới đây là một số cách phổ biến để tổ chức việc tạo ra thông tin thông minh:

- Kỹ thuật tách đặc trưng;
- Thực hiện tìm kiếm nhiều mô hình.
- Đuổi theo những sai lầm;
- Sử dụng siêu mô hình.
- Thực hiện giải trình tự mô hình.
- Bối cảnh phân vùng.
- Sử dụng ghi đè.

12. Thiết lập quy trình sáng tạo thông tin.

Hãy thiết lập nó để sử dụng tất cả các công cụ và dữ liệu có sẵn nhằm tạo ra và cải thiện trí thông minh. Điều này liên quan đến:

- Lựa chọn thể hiện cho trí thông minh;
- Tạo các phương pháp phỏng đoán đơn giản làm cơ sở.
- Sử dụng học máy để tạo ra trí thông minh phức tạp hơn khi cần thiết;
- Giúp hiểu được sự cân bằng giữa các phương án thực hiện khác nhau.
- Và đánh giá, lặp đi lặp lại, miễn là trí thông minh cần được cải thiện.

6.5.1.5. Điều phối hệ thống thông minh

Cần kiểm soát tất cả các công cụ được thảo luận và *lái xe đua* để đạt được thành công ngày này qua ngày khác miễn là hệ thống thông minh còn phù hợp. Một hệ thống thông minh được tổ chức tốt sẽ:

- Đạt được mục tiêu một cách đáng tin cậy theo thời gian.
- Có kinh nghiệm, trí tuệ và khách quan một cách cân bằng.
- Giảm thiểu sai sót một cách hiệu quả.
- Mở rộng quy mô một cách hiệu quả theo thời gian.
- Suy thoái chậm.

Việc điều phối rất quan trọng đối với hệ thống thông minh khi:

- Mục tiêu thay đổi.
- Người dùng thay đổi.
- Vấn đề thay đổi.
- Trí tuệ thay đổi.
- Hệ thống cần hiệu quả hơn.
- Sự lạm dụng xảy ra.

13. Lập kế hoạch điều phối trong suốt vòng đời của hệ thống thông minh.

Kế hoạch này bao gồm cách bắt đầu và cách đầu tư theo thời gian. Để điều phối một hệ thống thông minh, cần phải

- Giám sát các tiêu chí thành công.
- Kiểm tra sự tương tác.
- Cân bằng trải nghiệm.
- Ghi đè thông tin thông minh.
- Tạo ra trí thông minh.

14. Chuẩn bị nhận diện và xử lý sai sót.

Vì có trí tuệ thì sai lầm sẽ đến. Cần xác định lý do tại sao hệ thống thông minh gặp lỗi, bao gồm:

- Hệ thống ngừng hoạt động.
- Mô hình ngừng hoạt động.
- Lỗi thông minh.
- Suy thoái trí tuệ.

Và khi xác định được nguồn gốc của sai lầm, phải quyết định cách phản ứng:

- Đầu tư vào trí tuệ.
- Cân bằng trải nghiệm.

- Rèn luyện trí thông minh một cách khác biệt.
- Triển khai lan can.
- Ghi đè lỗi.

Bằng cách chấp nhận sai sót, chủ động tìm ra chúng và giảm thiểu thiệt hại mà chúng gây ra, có thể giải phóng nhiều không gian để trí tuệ tỏa sáng.

15. Hãy sẵn sàng với lạm dụng.

Nếu ai đó có thể tìm ra cách kiếm tiền bằng cách lạm dụng người dùng hoặc hệ thống thông minh, thì họ sẽ làm được. Vậy nên chuẩn bị để nhanh chóng xác định tình trạng lạm dụng nếu nó bắt đầu xảy ra, bằng cách tìm kiếm:

- Nhóm người dùng có mức sử dụng suy thoái.
- Các bối cảnh cho thấy hoạt động tăng đột biến.
- Bối cảnh cho thấy những thay đổi mạnh mẽ về kết quả.
- Các hình thức khiếu nại và báo cáo vấn đề.

Nếu lạm dụng xảy ra, hãy nhớ rằng đó là một hoạt động kinh doanh. không cần phải chặn hành vi lạm dụng. Chỉ cần làm cho kẻ bạo hành nghĩ rằng họ không thể kiếm tiền và họ sẽ dừng lại. Dưới đây là một số cách có thể làm điều này:

- Thêm chi phí cho người dùng hệ thống thông minh.
- Trở nên kém thú vị hơn đối với những kẻ lạm dụng.
- Làm mẫu với đối thủ.
- Loại bỏ đối thủ khỏi vòng lặp.

16. Tận hưởng chuyến đi.

Và đó không phải là kết thúc mà là sự khởi đầu. hệ thống thông minh có thể tồn tại trong nhiều năm.

6.5.2. Tóm tắt

Với 16 ý chính trên, đến đây đã có kiến thức để tiếp cận một dự án hệ thống thông minh một cách tự tin.

6.6. Kết luận

Chương 7

Thực hiện hệ thống thông minh

7.1. Giới thiệu

7.2. Phân tích kinh doanh

7.2.1. Tổ chức kinh doanh

7.2.1.1. Doanh nghiệp

Thuật ngữ kinh doanh đề cập đến một tổ chức hoặc thực thể dám nghĩ dám làm tham gia vào các hoạt động thương mại, công nghiệp hoặc chuyên nghiệp. Mục đích của doanh nghiệp là tổ chức một số loại hình sản xuất kinh tế hàng hóa hoặc dịch vụ. Các doanh nghiệp có thể là các tổ chức vì lợi nhuận hoặc các tổ chức phi lợi nhuận thực hiện sứ mệnh từ thiện hoặc thúc đẩy sự nghiệp xã hội. Các doanh nghiệp có quy mô và phạm vi từ doanh nghiệp tư nhân đến các tập đoàn quốc tế lớn.

Thuật ngữ kinh doanh cũng đề cập đến những nỗ lực và hoạt động được thực hiện bởi các cá nhân để sản xuất và bán hàng hóa và dịch vụ nhằm kiếm lợi nhuận.

7.2.1.2. Ích lợi của doanh nghiệp

- Doanh nghiệp được định nghĩa là một tổ chức hoặc thực thể kinh doanh tham gia vào các hoạt động thương mại, công nghiệp hoặc chuyên môn.
- Doanh nghiệp có thể là tổ chức vì lợi nhuận hoặc tổ chức phi lợi nhuận.
- Các loại hình kinh doanh bao gồm từ công ty trách nhiệm hữu hạn đến công ty tư nhân, công ty và công ty hợp danh.
- Một số doanh nghiệp hoạt động với quy mô nhỏ trong một ngành duy nhất trong khi những doanh nghiệp khác hoạt động với quy mô lớn trải rộng trên nhiều ngành trên thế giới.
- Apple và Walmart là hai ví dụ điển hình về những doanh nghiệp thành công và nổi tiếng.

7.2.1.3. Hiểu một doanh nghiệp

Thuật ngữ kinh doanh thường đề cập đến một thực thể hoạt động vì lý do thương mại, công nghiệp hoặc chuyên nghiệp. Khái niệm này bắt đầu bằng một ý tưởng và một cái tên, đồng thời có thể cần phải nghiên cứu thị trường sâu rộng để xác định mức độ khả thi khi biến ý tưởng đó thành một hoạt động kinh doanh.

Các doanh nghiệp thường yêu cầu kế hoạch kinh doanh trước khi bắt đầu hoạt động. Kế hoạch kinh doanh là một tài liệu chính thức phác thảo các mục tiêu và mục tiêu của công ty, đồng thời liệt kê các chiến lược và kế hoạch để đạt được các mục tiêu và mục tiêu này. Kế hoạch kinh doanh là điều cần thiết khi muốn vay vốn để bắt đầu hoạt động.

Xác định cơ cấu pháp lý của doanh nghiệp là một yếu tố quan trọng cần xem xét, vì chủ doanh nghiệp có thể cần phải có giấy phép và giấy phép cũng như tuân theo các yêu cầu đăng ký để bắt đầu hoạt động hợp pháp. Các tập đoàn được coi là pháp nhân ở nhiều quốc gia, có nghĩa là doanh nghiệp có thể sở hữu tài sản, nhận nợ và bị kiện ra tòa.

Một cái tên hay thường là một trong những tài sản quý giá nhất của doanh nghiệp, vì vậy điều quan trọng là chủ doanh nghiệp phải chọn tên một cách khôn ngoan.

Hầu hết các doanh nghiệp hoạt động để tạo ra lợi nhuận, thường được gọi là vì lợi nhuận. Tuy nhiên, một số doanh nghiệp có mục tiêu thúc đẩy một mục đích nhất định mà không mang lại lợi nhuận được gọi là phi lợi nhuận hoặc phi lợi nhuận. Các thực thể này có thể hoạt động như các tổ chức từ thiện, nghệ thuật, văn hóa, giáo dục và giải trí, các nhóm chính trị và vận động hoặc các tổ chức dịch vụ xã hội.

Hoạt động kinh doanh thường bao gồm việc mua bán hàng hóa và dịch vụ. Hoạt động kinh doanh có thể diễn ra ở bất cứ đâu, cho dù đó là ở cửa hàng thực tế, trực tuyến hay ven đường. Bất kỳ ai tiến hành hoạt động kinh doanh có thu nhập tài chính đều phải báo cáo thu nhập này cho Sở Thuế Vụ (IRS).

Một công ty thường xác định hoạt động kinh doanh của mình theo ngành mà nó hoạt động. Ví dụ: kinh doanh bất động sản, kinh doanh quảng cáo hoặc kinh doanh sản xuất nệm là những ví dụ về ngành công nghiệp.

Kinh doanh là thuật ngữ thường được sử dụng để chỉ các giao dịch liên quan đến sản phẩm hoặc dịch vụ cơ bản. Ví dụ: ExxonMobil tiến hành hoạt động kinh doanh của mình bằng cách cung cấp dầu.

7.2.2. Kiểu doanh nghiệp

Các loại hình doanh nghiệp: Có nhiều cách để tổ chức kinh doanh và có nhiều cấu trúc pháp lý và thuế khác nhau tương ứng với từng cách. Các doanh nghiệp thường được phân loại và có cấu trúc chung như sau:

- Doanh nghiệp tư nhân: Như tên gọi, doanh nghiệp tư nhân được sở hữu và điều hành bởi một người duy nhất. Không có sự tách biệt về mặt pháp lý giữa doanh nghiệp và chủ sở hữu, điều đó có nghĩa là trách nhiệm pháp lý và thuế của doanh nghiệp là trách nhiệm của chủ sở hữu.
- Quan hệ đối tác: Quan hệ đối tác là mối quan hệ kinh doanh giữa hai hoặc nhiều người cùng nhau tiến hành kinh doanh. Mỗi đối tác đóng góp nguồn lực và tiền bạc cho doanh nghiệp và chia sẻ lợi nhuận và thua lỗ của doanh nghiệp. Lợi nhuận và thua lỗ được chia sẻ được ghi nhận trên tờ khai thuế của mỗi đối tác.
- Công ty: Công ty là một doanh nghiệp trong đó một nhóm người hoạt động như một thực thể duy nhất. Chủ sở hữu thường được gọi là cổ đông trao đổi quyền lợi để lấy cổ phiếu phổ thông của công ty. Việc thành lập một doanh nghiệp sẽ giải phóng chủ sở hữu trách nhiệm tài chính đối với các nghĩa vụ kinh doanh. Một công ty có những quy định về thuế bất lợi cho chủ sở hữu doanh nghiệp.
- Công ty trách nhiệm hữu hạn (LLC): Đây là một loại hình kinh doanh tương đối mới và lần đầu tiên xuất hiện ở Wyoming vào năm 1977 và ở các tiểu bang khác vào những năm 1990. Công ty trách nhiệm hữu hạn kết hợp lợi ích về thuế thông qua của công ty hợp danh với lợi ích trách nhiệm hữu hạn của một công ty.

7.2.3. Quy mô doanh nghiệp

7.2.3.1. Các doanh nghiệp nhỏ

Các công ty nhỏ do chủ sở hữu điều hành được gọi là doanh nghiệp nhỏ. Thường được quản lý bởi một người hoặc một nhóm nhỏ với ít hơn 100 nhân viên, các công ty này bao gồm các nhà hàng gia đình, công ty kinh

doanh tại nhà, công ty quần áo, sách và xuất bản cũng như các nhà sản xuất nhỏ. Tính đến năm 2021, có 33,2 triệu doanh nghiệp nhỏ ở Hoa Kỳ với 61,7 triệu nhân viên đang hoạt động.

Cơ quan quản lý doanh nghiệp nhỏ (SBA) sử dụng số lượng nhân viên làm việc tại một công ty và doanh thu hàng năm của công ty để chính thức xác định một doanh nghiệp nhỏ. Đối với 229 lĩnh vực công nghiệp, từ kỹ thuật và sản xuất đến dịch vụ thực phẩm và bất động sản, SBA đặt ra tiêu chuẩn định cỡ 5 năm một lần.

Các doanh nghiệp đáp ứng các tiêu chuẩn của SBA có thể đủ điều kiện nhận các khoản vay, trợ cấp và các hợp đồng "dành riêng cho doanh nghiệp nhỏ", trong đó chính phủ liên bang hạn chế cạnh tranh để giúp các doanh nghiệp nhỏ cạnh tranh và giành được hợp đồng liên bang.

7.2.3.2. Doanh nghiệp vừa

Không có thông số kỹ thuật dứt khoát ở Hoa Kỳ để xác định một công ty cỡ trung bình hoặc cỡ trung bình. Tuy nhiên, khi các thành phố lớn của Hoa Kỳ như Philadelphia, Baltimore và Boston đánh giá bối cảnh hoạt động kinh doanh, một công ty cỡ trung bình được xác định là một công ty có từ 100 đến 249 nhân viên hoặc tổng doanh thu hàng năm từ 10 triệu USD đến dưới 1 tỷ USD.

7.2.3.3. Doanh nghiệp lớn

Các doanh nghiệp lớn thường có từ 250 nhân viên trở lên và thu được tổng doanh thu hơn 1 tỷ USD. Họ có thể phát hành cổ phiếu doanh nghiệp để tài trợ cho hoạt động với tư cách là một công ty giao dịch công khai.

Các doanh nghiệp lớn có thể có trụ sở tại một quốc gia và hoạt động quốc tế. Chúng thường được tổ chức bởi các phòng ban, chẳng hạn như nhân sự, tài chính, tiếp thị, bán hàng, nghiên cứu và phát triển.

Không giống như các doanh nghiệp vừa và nhỏ, thuộc sở hữu của một cá nhân hoặc một nhóm người, các tổ chức lớn thường tách gánh nặng thuế của họ khỏi chủ sở hữu, những người thường không quản lý công ty của mình mà thay vào đó, một ban giám đốc được bầu ra sẽ ban hành hầu hết các quyết định kinh doanh.

7.2.4. Phân tích dữ liệu

7.2.4.1. Khái niệm phân tích dữ liệu

Định nghĩa : Phân tích dữ liệu (Data analysis) là phương pháp làm việc với dữ liệu để thu thập thông tin hữu ích, sau đó có thể được sử dụng để đưa ra quyết định sáng suốt.

Sherlock Holme tuyên bố trong cuốn A Scandal in Bohemia của Sir Arthur Conan Doyle: “Sẽ là một sai lầm lớn khi đưa ra giả thuyết trước khi có dữ liệu. Một cách vô cảm, người ta bắt đầu bóp méo sự thật cho phù hợp với lý thuyết, thay vì lý thuyết cho phù hợp với sự thật”.

Ý tưởng này nằm ở gốc rễ của việc phân tích dữ liệu. Khi chúng ta có thể trích xuất ý nghĩa từ dữ liệu, nó sẽ giúp chúng ta đưa ra quyết định tốt hơn. Và chúng ta đang sống trong thời đại mà chúng ta có nhiều dữ liệu hơn bao giờ hết trong tầm tay.

Các công ty đang nhận thức được lợi ích của việc tận dụng dữ liệu. Phân tích dữ liệu có thể giúp ngân hàng cá nhân hóa các tương tác của khách hàng, hệ thống chăm sóc sức khỏe để dự đoán nhu cầu sức khỏe trong tương lai hoặc một công ty giải trí tạo ra hit lớn tiếp theo.

Báo cáo Tương lai việc làm năm 2023 của Diễn đàn Kinh tế Thế giới đã liệt kê các nhà phân tích dữ liệu và nhà khoa học là một trong những công việc có nhu cầu cao nhất, bên cạnh các chuyên gia về AI và máy học cũng như chuyên gia về dữ liệu lớn. Ở đây sẽ tìm hiểu thêm về quy trình phân tích dữ liệu, các loại phân tích dữ liệu khác nhau và các khóa học được đề xuất để giúp bắt đầu trong lĩnh vực này.

7.2.4.2. Quá trình phân tích dữ liệu

Khi dữ liệu có sẵn cho các công ty tiếp tục tăng cả về số lượng và độ phức tạp, thì nhu cầu về một quy trình hiệu quả và hiệu suất để khai thác giá trị của dữ liệu đó cũng tăng theo. Quá trình phân tích dữ liệu thường trải qua một số giai đoạn lặp đi lặp lại. Chúng ta hãy xem xét kỹ hơn về từng cái.

- Xác định câu hỏi kinh doanh mà người ta muốn trả lời. Công ty đang cố gắng giải quyết vấn đề gì? cần đo lường những gì và sẽ đo lường nó như thế nào?
- Thu thập các tập dữ liệu thô cần để trả lời câu hỏi đã xác định. Việc thu thập dữ liệu có thể đến từ các nguồn nội bộ, như phần mềm quản lý quan hệ khách hàng (CRM) của công ty hoặc từ các nguồn

thứ cấp, như hồ sơ chính phủ hoặc giao diện lập trình ứng dụng truyền thông xã hội (API).

- Làm sạch dữ liệu để chuẩn bị phân tích. Điều này thường liên quan đến việc xóa dữ liệu trùng lặp và bất thường, điều hòa sự không nhất quán, chuẩn hóa cấu trúc và định dạng dữ liệu cũng như xử lý khoảng trắng và các lỗi cú pháp khác.
- Phân tích dữ liệu. Bằng cách xử lý dữ liệu bằng các kỹ thuật và công cụ phân tích dữ liệu khác nhau, có thể bắt đầu tìm ra xu hướng, mối tương quan, ngoại lệ và các biến thể kể một câu chuyện. Trong giai đoạn này, có thể sử dụng khai thác dữ liệu để khám phá các mẫu trong cơ sở dữ liệu hoặc phần mềm trực quan hóa dữ liệu để giúp chuyển đổi dữ liệu thành định dạng đồ họa dễ hiểu.
- Giải thích kết quả phân tích để xem dữ liệu trả lời câu hỏi ban đầu tốt đến mức nào. Có thể đưa ra khuyến nghị gì dựa trên dữ liệu? Những hạn chế đối với kết luận là gì?

7.2.5. Các loại phân tích dữ liệu

Dữ liệu có thể được sử dụng để trả lời các câu hỏi và hỗ trợ các quyết định theo nhiều cách khác nhau. Để xác định cách tốt nhất để phân tích ngày tháng, có thể tự làm quen với bốn loại phân tích dữ liệu thường được sử dụng trong hiện trường.

7.2.5.1. Phân tích mô tả

Phân tích mô tả[1] cho chúng ta biết điều gì đã xảy ra. Loại phân tích này giúp mô tả hoặc tóm tắt dữ liệu định lượng bằng cách trình bày số liệu thống kê. Ví dụ: phân tích thống kê mô tả có thể hiển thị sự phân bổ doanh số bán hàng trên một nhóm nhân viên và con số bán hàng trung bình trên mỗi nhân viên.

7.2.5.2. Phân tích chẩn đoán

Nếu phân tích mô tả xác định “cái gì” thì phân tích chẩn đoán[2] sẽ xác định “tại sao”. Giả sử một phân tích mô tả cho thấy một lượng bệnh nhân bất thường trong bệnh viện. Đi sâu hơn vào dữ liệu có thể tiết lộ rằng nhiều bệnh

[1] Descriptive analysis: phân tích mô tả
[2] Diagnostic analysis : phân tích chẩn đoán

nhân trong số này có chung các triệu chứng của một loại vi-rút cụ thể. Phân tích chẩn đoán này có thể giúp xác định rằng tác nhân lây nhiễm đã dẫn đến lượng bệnh nhân đổ vào, tức là trả lời cho câu hỏi tại sao.

7.2.5.3. Phân tích dự đoán

Cho đến nay, chúng ta đã xem xét các loại phân tích nhằm kiểm tra và rút ra kết luận về quá khứ. Phân tích dự đoán[1] sử dụng dữ liệu để hình thành các dự đoán về tương lai. Bằng cách sử dụng phân tích dự đoán, có thể nhận thấy rằng một sản phẩm nhất định có doanh số bán hàng tốt nhất trong tháng 9 và tháng 10 hàng năm, khiến người ta dự đoán mức cao tương tự trong năm tới.

7.2.5.4. Phân tích theo quy định

Phân tích theo quy định[2] lấy tất cả những hiểu biết sâu sắc thu thập được từ ba loại phân tích đầu tiên và sử dụng chúng để đưa ra khuyến nghị về cách một công ty nên hành động. Sử dụng ví dụ trước của chúng tôi, loại phân tích này có thể đề xuất một kế hoạch thị trường nhằm phát triển dựa trên sự thành công của những tháng doanh số bán hàng cao và khai thác các cơ hội tăng trưởng mới trong những tháng có doanh số thấp hơn.

Ra quyết định dựa trên dữ liệu (DDDM)

Việc ra quyết định dựa trên dữ liệu, đôi khi được viết tắt là DDDM[3], có thể được định nghĩa là quá trình đưa ra quyết định kinh doanh chiến lược dựa trên sự kiện, dữ liệu và số liệu thay vì trực giác, cảm xúc hoặc quan sát.

Điều này nghe có vẻ hiển nhiên, nhưng trên thực tế, không phải tất cả các tổ chức đều dựa vào dữ liệu như họ có thể. Theo công ty tư vấn quản lý toàn cầu McKinsey Global Institute, các công ty dựa trên dữ liệu sẽ có khả năng thu hút khách hàng mới tốt hơn, duy trì lòng trung thành của khách hàng và đạt được lợi nhuận trên mức trung bình.

[1] Predictive analysis : phân tích dự đoán
[2] Prescriptive analysis: phân tích theo qui định
[3] data-driven decision-making

7.2.6. Phân tích kinh doanh

Định nghĩa: Phân tích kinh doanh (Business analytics) là công việc đề cập đến các phương pháp thống kê và công nghệ điện toán để xử lý, khai thác và trực quan hóa dữ liệu nhằm khám phá các mô hình, mối quan hệ và hiểu biết sâu sắc giúp đưa ra quyết định kinh doanh tốt hơn.

Phân tích kinh doanh liên quan đến việc các công ty sử dụng dữ liệu do hoạt động của họ tạo ra hoặc dữ liệu có sẵn công khai để giải quyết các vấn đề kinh doanh, giám sát các nguyên tắc kinh doanh cơ bản của họ, xác định các cơ hội tăng trưởng mới và phục vụ khách hàng tốt hơn.

Phân tích kinh doanh sử dụng tính năng khám phá dữ liệu, trực quan hóa dữ liệu, bảng thông tin tích hợp, v.v. để cung cấp cho người dùng quyền truy cập vào dữ liệu hữu ích và thông tin chi tiết về doanh nghiệp.

7.2.6.1. Phân tích kinh doanh so với kinh doanh thông minh

Thông minh doanh nghiệp BI[1] cho phép đưa ra các quyết định kinh doanh tốt hơn dựa trên nền tảng dữ liệu kinh doanh. Phân tích kinh doanh là một tập hợp con của thông tin kinh doanh, trong đó phân tích kinh doanh cung cấp phân tích, trong khi cơ sở hạ tầng thông minh kinh doanh bao trùm bao gồm các công cụ để xác định và lưu trữ dữ liệu sẽ được sử dụng để ra quyết định.

BI thu thập, quản lý và sử dụng cả dữ liệu đầu vào thô cũng như kiến thức thu được và hiểu biết sâu sắc có thể hành động được tạo ra bởi phân tích kinh doanh. Mục đích liên tục của phân tích kinh doanh là phát triển kiến thức và hiểu biết mới để nâng cao tổng thể trí tuệ kinh doanh của công ty.

Phân tích kinh doanh có thể được sử dụng để trả lời các câu hỏi về những gì đã xảy ra trong quá khứ, đưa ra dự đoán và dự báo kết quả kinh doanh. Một tổ chức có thể có được bức tranh hoàn chỉnh hơn về hoạt động kinh doanh của mình, cho phép tổ chức hiểu được hành vi của người dùng một cách hiệu quả hơn.

Các nhà khoa học dữ liệu và nhà phân tích dữ liệu nâng cao sử dụng phân tích kinh doanh để cung cấp phân tích thống kê nâng cao. Một số ví dụ về phân tích thống kê bao gồm phân tích hồi quy sử dụng dữ liệu bán hàng trước đó để ước tính giá trị vòng đời của khách hàng và phân tích cụm để

[1] Business intelligence (BI): thông minh doanh nghiệp

phân tích và phân khúc người dùng sử dụng nhiều và ít sử dụng trong một khu vực cụ thể.

Giải pháp phân tích kinh doanh mang lại lợi ích cho tất cả các bộ phận, bao gồm tài chính, nhân sự, chuỗi cung ứng, tiếp thị, bán hàng hoặc công nghệ thông tin, cùng với tất cả các ngành, bao gồm chăm sóc sức khỏe, dịch vụ tài chính và hàng tiêu dùng.

7.2.6.2. Phương pháp phân tích kinh doanh

Phân tích kinh doanh sử dụng phân tích, tức hành động thu thập thông tin chi tiết từ dữ liệu, để thúc đẩy tăng hiệu suất kinh doanh.

Như với phân tích dữ liệu, có bốn loại phân tích có giá trị thường được sử dụng:

1. *Phân tích mô tả:* Đúng như tên gọi, loại phân tích này mô tả dữ liệu mà nó chứa. Một ví dụ là biểu đồ hình tròn phân tích thông tin nhân khẩu học của khách hàng của công ty.
2. *Phân tích chẩn đoán*: Phân tích chẩn đoán giúp xác định nguyên nhân gốc rễ của một sự kiện. Nó có thể giúp trả lời các câu hỏi như: Chuỗi sự kiện nào ảnh hưởng đến kết quả kinh doanh? Mối tương quan thực sự và quan hệ nhân quả nằm ở đâu trong một khung thời gian lịch sử nhất định? Động lực đằng sau những phát hiện này là gì? Ví dụ: nhà sản xuất có thể phân tích một bộ phận bị lỗi trên dây chuyền lắp ráp và xác định nguyên nhân gây ra lỗi đó.
3. *Phân tích dự đoán*: Phân tích dự đoán khai thác dữ liệu hiện có, xác định các mẫu và giúp các công ty dự đoán những gì có thể xảy ra trong tương lai dựa trên dữ liệu đó. Nó sử dụng các mô hình dự đoán để đưa ra các giả thuyết về hành vi hoặc kết quả trong tương lai. Ví dụ: một tổ chức có thể đưa ra dự đoán về sự thay đổi trong doanh số bán áo khoác nếu mùa đông sắp tới được dự đoán có nhiệt độ ấm hơn.

   Mô hình dự đoán cũng giúp các tổ chức tránh được các vấn đề trước khi chúng xảy ra, chẳng hạn như biết khi nào một phương tiện hoặc công cụ sẽ hỏng và can thiệp trước khi nó xảy ra hoặc biết khi nào việc thay đổi về nhân khẩu học hoặc tâm lý học sẽ tác động tích cực hoặc tiêu cực đến dòng sản phẩm của họ.

4. *Phân tích theo quy định:* Những phân tích này giúp các tổ chức đưa ra quyết định về tương lai dựa trên thông tin và tài nguyên hiện có. Mọi doanh nghiệp đều có thể sử dụng phân tích theo quy định bằng cách xem xét dữ liệu hiện có của họ để đoán xem điều gì sẽ xảy ra tiếp theo. Ví dụ: các tổ chức tiếp thị và bán hàng có thể phân tích tỷ lệ khách hàng tiềm năng thành công của nội dung gần đây để xác định loại nội dung nào họ nên ưu tiên trong tương lai. Các công ty dịch vụ tài chính sử dụng nó để phát hiện gian lận bằng cách phân tích dữ liệu hiện có để đưa ra quyết định theo thời gian thực về việc liệu bất kỳ giao dịch mua hàng nào có khả năng gian lận hay không.

7.2.6.3. Các công cụ và kỹ thuật phân tích kinh doanh

Thực tiễn phân tích kinh doanh bao gồm một số công cụ giúp các công ty hiểu được dữ liệu họ đang thu thập và sử dụng dữ liệu đó để biến dữ liệu đó thành thông tin chuyên sâu. Dưới đây là một số công cụ, nguyên tắc và cách tiếp cận phổ biến nhất:

- *Quản lý dữ liệu*: Quản lý dữ liệu là hoạt động thu thập, xử lý, bảo mật và lưu trữ dữ liệu của tổ chức. Sau đó, nó được sử dụng để đưa ra quyết định chiến lược nhằm cải thiện kết quả kinh doanh. Kỷ luật quản lý dữ liệu ngày càng trở thành ưu tiên khi việc mở rộng kho dữ liệu đã tạo ra những thách thức đáng kể, chẳng hạn như kho dữ liệu, rủi ro bảo mật và tắc nghẽn chung trong việc ra quyết định.
- *Khai thác dữ liệu hay KDD*: Khai thác dữ liệu, còn được gọi là khám phá tri thức trong dữ liệu (KDD[1]), là quá trình khám phá các mẫu và thông tin có giá trị khác từ các tập dữ liệu lớn và là một thành phần quan trọng của phân tích dữ liệu lớn. Tầm quan trọng ngày càng tăng của dữ liệu lớn khiến việc khai thác dữ liệu trở thành một thành phần quan trọng của bất kỳ doanh nghiệp hiện đại nào bằng cách hỗ trợ các công ty chuyển đổi dữ liệu thô thành kiến thức hữu ích.

[1] Knowledge discovery in data KDD: khám phá tri thức trong dữ liệu

- *Kho dữ liệu*: Kho dữ liệu hay kho dữ liệu doanh nghiệp (EDW[1]) là một hệ thống tổng hợp dữ liệu từ nhiều nguồn khác nhau, bao gồm ứng dụng, thiết bị mạng vạn vật IoT[2], phương tiện truyền thông xã hội và bảng tính vào một kho dữ liệu nhất quán, tập trung và duy nhất để hỗ trợ phân tích dữ liệu, khai thác dữ liệu, trí tuệ nhân tạo (AI) và học máy (ML). Hệ thống kho dữ liệu cho phép tổ chức chạy các phân tích mạnh mẽ trên lượng lớn dữ liệu (petabyte và petabyte) theo cách mà cơ sở dữ liệu tiêu chuẩn không thể làm được.
- *Trực quan hóa dữ liệu*: Việc trình bày dữ liệu bằng cách sử dụng đồ họa như biểu đồ, sơ đồ, đồ họa thông tin và thậm chí cả hình ảnh động. Những màn hình hiển thị thông tin trực quan này truyền đạt các mối quan hệ dữ liệu phức tạp và thông tin chi tiết dựa trên dữ liệu theo cách dễ hiểu hơn, đặc biệt hữu ích cho những nhân viên không chuyên về kỹ thuật hiểu các khái niệm phân tích và giúp hiển thị các mẫu trong nhiều điểm dữ liệu. Trực quan hóa dữ liệu cũng có thể giúp tạo ra ý tưởng, minh họa ý tưởng hoặc khám phá trực quan.
- *Dự báo*: Công cụ này lấy dữ liệu lịch sử và điều kiện thị trường hiện tại, sau đó đưa ra dự đoán về mức doanh thu mà một tổ chức có thể mong đợi mang lại trong vài tháng hoặc năm tới. Dự báo được điều chỉnh khi có thông tin mới. Khi các công ty sử dụng dữ liệu và phân tích bằng các phương pháp hay nhất về lập kế hoạch và dự báo được thiết lập tốt, họ sẽ nâng cao khả năng ra quyết định chiến lược và có thể được khen thưởng bằng các kế hoạch chính xác hơn và dự báo kịp thời hơn.
- *Thuật toán học máy*: Thuật toán học máy là một tập hợp các quy tắc hoặc quy trình được hệ thống AI sử dụng để thực hiện các tác vụ, thường là để khám phá các mẫu và thông tin chuyên sâu về dữ liệu mới hoặc để dự đoán giá trị đầu ra từ một tập hợp các biến đầu vào nhất định. Các thuật toán học máy cho phép học máy học, cung cấp khả năng phân tích dữ liệu, xác định xu hướng và dự đoán các vấn đề trước khi chúng xảy ra.

[1] Enterprise data warehouse EDW: kho dữ liệu doanh nghiệp
[2] Internet of Things (IoT): mạng vạn vật

- *Báo cáo:* Phân tích kinh doanh hoạt động dựa trên dữ liệu để giúp các tổ chức đưa ra quyết định sáng suốt. Phần mềm báo cáo cấp doanh nghiệp có thể trích xuất thông tin từ nhiều ứng dụng khác nhau được doanh nghiệp sử dụng, phân tích dữ liệu và tạo báo cáo.
- *Phân tích thống kê*: Phân tích thống kê cho phép tổ chức rút ra những hiểu biết sâu sắc có thể hành động từ dữ liệu của mình. Quy trình phân tích thống kê nâng cao giúp đảm bảo độ chính xác cao và đưa ra quyết định có chất lượng. Vòng đời phân tích bao gồm việc chuẩn bị và quản lý dữ liệu cho đến phân tích và báo cáo.
- *Phân tích văn bản*: Xác định các mẫu và xu hướng văn bản trong dữ liệu phi cấu trúc bằng cách sử dụng máy học, thống kê và ngôn ngữ học. Bằng cách chuyển đổi dữ liệu sang định dạng có cấu trúc hơn thông qua khai thác văn bản và phân tích văn bản, có thể tìm thấy nhiều hiểu biết định lượng hơn.

7.2.6.4. Lợi ích của phân tích kinh doanh

Các tổ chức hiện đại cần có khả năng đưa ra quyết định nhanh chóng để cạnh tranh trong một thế giới đang thay đổi nhanh chóng, nơi các đối thủ cạnh tranh mới thường xuyên xuất hiện và thói quen của khách hàng luôn thay đổi. Các tổ chức ưu tiên phân tích kinh doanh có một số lợi thế so với các đối thủ cạnh tranh không ưu tiên.

Quyết định nhanh hơn và sáng suốt hơn: Việc có cái nhìn linh hoạt và mở rộng về tất cả dữ liệu mà tổ chức sở hữu có thể loại bỏ sự không chắc chắn, thúc đẩy tổ chức hành động nhanh hơn và cải thiện quy trình kinh doanh. Nếu dữ liệu của một tổ chức cho thấy doanh số của một dòng sản phẩm cụ thể đang giảm nhanh chóng thì tổ chức đó có thể quyết định ngừng sản xuất dòng sản phẩm đó. Nếu rủi ro khí hậu ảnh hưởng đến việc thu hoạch nguyên liệu thô mà tổ chức khác phụ thuộc vào, thì tổ chức đó có thể cần phải tìm nguồn nguyên liệu mới từ nơi khác. Nó đặc biệt hữu ích khi xem xét chiến lược giá cả.

Cách một công ty định giá hàng hóa hoặc dịch vụ của mình dựa trên hàng nghìn điểm dữ liệu, nhiều điểm trong số đó không đứng yên theo thời gian. Cho dù công ty có chiến lược định giá cố định hay động, việc có thể truy cập dữ liệu thời gian thực để tạo ra dữ liệu định giá ngắn hạn và dài hạn

thông minh hơn là rất quan trọng. Đối với các tổ chức muốn kết hợp tính năng định giá linh hoạt, phân tích kinh doanh cho phép họ sử dụng hàng nghìn điểm dữ liệu để phản ứng với các sự kiện và xu hướng bên ngoài nhằm xác định mức giá có lợi nhất một cách thường xuyên nếu cần.

Chế độ xem thông tin một cửa sổ: Sự cộng tác ngày càng tăng giữa các phòng ban và người dùng trong ngành kinh doanh có nghĩa là mọi người đều có cùng một dữ liệu và đang nói từ cùng một cẩm nang. Việc có một tấm kính duy nhất đó sẽ hiển thị nhiều mô hình chưa được nhìn thấy hơn, cho phép các bộ phận khác nhau hiểu được cách tiếp cận toàn diện của công ty và tăng khả năng phản ứng của tổ chức với những thay đổi trên thị trường.

Nâng cao dịch vụ khách hàng: Bằng cách biết khách hàng muốn gì, khi nào và như thế nào, các tổ chức sẽ khuyến khích khách hàng hài lòng hơn và xây dựng lòng trung thành lớn hơn. Ngoài trải nghiệm khách hàng được cải thiện, nhờ có thể đưa ra quyết định thông minh hơn về phân bổ nguồn lực hoặc sản xuất, các tổ chức có thể cung cấp những hàng hóa hoặc dịch vụ đó với mức giá phải chăng hơn.

7.2.6.5. Vai trò trong phân tích kinh doanh

Các công ty muốn khai thác dữ liệu kinh doanh có thể sẽ cần nâng cao kỹ năng của nhân viên hiện có hoặc thuê nhân viên mới, có khả năng tạo ra các mô tả công việc mới. Các tổ chức dựa trên dữ liệu cần nhân viên có kỹ năng giao tiếp và phân tích thực hành xuất sắc.

Dưới đây là một số nhân viên mà họ cần để tận dụng tối đa tiềm năng của các chiến lược phân tích kinh doanh mạnh mẽ:

Nhà khoa học dữ liệu: Những người này chịu trách nhiệm quản lý các thuật toán và mô hình hỗ trợ các chương trình phân tích kinh doanh. Các nhà khoa học dữ liệu của tổ chức sử dụng các thư viện nguồn mở, chẳng hạn như bộ công cụ ngôn ngữ tự nhiên (NTLK) cho các thuật toán hoặc tự xây dựng các thư viện để phân tích dữ liệu. Họ xuất sắc trong việc giải quyết vấn đề và thường cần biết một số ngôn ngữ lập trình, chẳng hạn như Python, giúp truy cập các thuật toán học máy sẵn có và ngôn ngữ truy vấn có cấu trúc (SQL), giúp trích xuất dữ liệu từ cơ sở dữ liệu để đưa vào hệ thống. người mẫu.

Trong những năm gần đây, ngày càng có nhiều trường cấp bằng Thạc sĩ Khoa học hoặc Cử nhân về khoa học dữ liệu, nơi sinh viên tham gia vào các khóa học của chương trình cấp bằng dạy họ về khoa học máy tính, mô hình thống kê và các ứng dụng toán học khác.

Kỹ sư dữ liệu: Họ tạo và duy trì các hệ thống thông tin thu thập dữ liệu từ những nơi khác nhau được làm sạch và sắp xếp rồi đưa vào cơ sở dữ liệu chính. Họ thường chịu trách nhiệm giúp đảm bảo rằng dữ liệu có thể được các bên liên quan dễ dàng thu thập và truy cập để cung cấp cho các tổ chức cái nhìn thống nhất về hoạt động dữ liệu của họ.

Các nhà phân tích dữ liệu: Họ đóng vai trò then chốt trong việc truyền đạt những hiểu biết sâu sắc cho các bên liên quan bên ngoài và bên trong. Tùy thuộc vào quy mô của tổ chức, họ có thể thu thập và phân tích các tập dữ liệu cũng như xây dựng trực quan hóa dữ liệu hoặc họ có thể đảm nhận công việc do các nhà khoa học dữ liệu khác tạo ra và tập trung vào việc xây dựng cách kể chuyện mạnh mẽ cho những điểm chính.

7.2.6.6. Các trường hợp sử dụng phân tích kinh doanh

Phân tích kinh doanh rất hữu ích cho mọi loại đơn vị kinh doanh như một cách để hiểu dữ liệu mà nó có và giúp nó tạo ra những hiểu biết cụ thể giúp thúc đẩy việc ra quyết định thông minh hơn.

- Lập kế hoạch tài chính và hoạt động: Phân tích kinh doanh cung cấp những hiểu biết có giá trị để giúp các tổ chức điều chỉnh kế hoạch tài chính và hoạt động của họ một cách liền mạch hơn. Nó thực hiện điều này bằng cách đặt ra các quy tắc quản lý chuỗi cung ứng, tích hợp dữ liệu giữa các chức năng và cải thiện khả năng phân tích chuỗi cung ứng cũng như dự báo nhu cầu.
- Phân tích lập kế hoạch: Một phương pháp lập kế hoạch kinh doanh tích hợp kết hợp bảng tính và công nghệ cơ sở dữ liệu để đưa ra quyết định kinh doanh hiệu quả về các chủ đề như nhu cầu và tạo khách hàng tiềm năng, tối ưu hóa chi phí vận hành và yêu cầu công nghệ dựa trên các số liệu vững chắc. Nhiều tổ chức trước đây đã sử dụng các công cụ bao gồm Microsoft Excel để lập kế hoạch kinh doanh, nhưng một số tổ chức đang chuyển sang các công cụ như IBM Planning Analytics.

- Lập kế hoạch tiếp thị và bán hàng tích hợp: Hầu hết các tổ chức đều có dữ liệu lịch sử về việc tạo khách hàng tiềm năng, chuyển đổi bán hàng và tỷ lệ giữ chân khách hàng thành công. Các tổ chức muốn tạo kế hoạch và dự báo doanh thu chính xác hơn cũng như có được tầm nhìn sâu hơn về dữ liệu tiếp thị và bán hàng của họ đang sử dụng phân tích kinh doanh để phân bổ nguồn lực dựa trên hiệu suất hoặc nhu cầu thay đổi nhằm đáp ứng các mục tiêu kinh doanh.
- Lập kế hoạch tích hợp hiệu quả hoạt động của lực lượng lao động: Khi các tổ chức trải qua quá trình chuyển đổi kỹ thuật số và phản ứng với bối cảnh thay đổi, họ có thể cần đảm bảo có lực lượng lao động phù hợp với các kỹ năng phân tích phù hợp. Điều này đặc biệt đúng trong một thế giới mà nhân viên có nhiều khả năng rời bỏ công ty để tìm công việc mới. Lập kế hoạch hiệu suất lực lượng lao động giúp các tổ chức hiểu được nhu cầu lực lượng lao động của họ.

7.2.7. Các bước phân tích kinh doanh

Để tối đa hóa lợi ích của việc phân tích kinh doanh của tổ chức, tổ chức cần phải làm sạch và kết nối dữ liệu của mình, tạo trực quan hóa dữ liệu và cung cấp thông tin chi tiết về vị trí hiện tại của doanh nghiệp đồng thời giúp dự đoán điều gì sẽ xảy ra vào ngày mai. Điều này thường bao gồm các bước sau:

7.2.7.1. Thu thập dữ liệu

Trước tiên, các tổ chức phải xác định tất cả dữ liệu họ có trong tay và dữ liệu bên ngoài nào họ muốn kết hợp để hiểu những cơ hội phân tích kinh doanh mà họ có.

7.2.7.2. Làm sạch dữ liệu

Thật không may, phần lớn dữ liệu của công ty vẫn chưa được làm sạch, khiến việc phân tích chính xác trở nên vô ích cho đến khi được xử lý.

Dưới đây là một số lý do tại sao dữ liệu của tổ chức có thể cần được làm sạch:

- Trường dữ liệu không chính xác: Do nhập thủ công hoặc truyền dữ liệu không chính xác, tổ chức có thể có dữ liệu xấu trộn lẫn với dữ liệu

chính xác. Nếu nó có bất kỳ dữ liệu xấu nào trong hệ thống, điều này có khả năng khiến toàn bộ tập hợp trở nên vô nghĩa.

- Giá trị dữ liệu lỗi thời: Một số bộ dữ liệu nhất định, bao gồm thông tin khách hàng, có thể cần chỉnh sửa do khách hàng rời đi, dòng sản phẩm bị ngừng sản xuất hoặc dữ liệu lịch sử khác không còn phù hợp.
- Thiếu dữ liệu: Các công ty có thể đã thay đổi cách họ thu thập dữ liệu hoặc dữ liệu họ thu thập, điều đó có nghĩa là các mục nhập lịch sử có thể thiếu dữ liệu quan trọng cho việc phân tích kinh doanh trong tương lai. Các công ty trong tình huống này có thể cần đầu tư vào việc nhập dữ liệu thủ công hoặc xác định cách sử dụng thuật toán hoặc học máy để dự đoán dữ liệu chính xác sẽ là gì.
- Kho dữ liệu: Nếu dữ liệu hiện có của tổ chức nằm trong nhiều bảng tính hoặc các loại cơ sở dữ liệu khác, thì tổ chức đó có thể cần phải hợp nhất dữ liệu để tất cả ở cùng một nơi. Mặc dù nền tảng của bất kỳ phương pháp phân tích kinh doanh nào đều là dữ liệu của bên thứ nhất (dữ liệu công ty đã thu thập từ các bên liên quan và sở hữu), nhưng họ có thể muốn nối thêm dữ liệu của bên thứ ba (dữ liệu họ đã mua hoặc thu thập được từ các tổ chức khác) để khớp với dữ liệu của họ. với những hiểu biết bên ngoài.

7.2.7.3. Phân tích dữ liệu

Giờ đây, các công ty có thể truy vấn và phân tích nhanh chóng hàng gigabyte hoặc terabyte dữ liệu với nhiều điện toán đám mây hơn. Các nhà khoa học dữ liệu có thể phân tích dữ liệu hiệu quả hơn bằng cách sử dụng máy học, thuật toán, trí tuệ nhân tạo (AI) và các công nghệ khác. Làm như vậy có thể tạo ra những hiểu biết sâu sắc có thể hành động dựa trên các chỉ số hiệu suất chính (KPI) của tổ chức.

7.2.7.4. Trực quan hóa dữ liệu

Giờ đây, các chương trình phân tích kinh doanh có thể nhanh chóng lấy lượng lớn dữ liệu được phân tích đó để tạo trang tổng quan, trực quan hóa và bảng nơi dữ liệu có thể được lưu trữ, xem, sắp xếp, thao tác và gửi cho các bên liên quan.

Các phương pháp hay nhất về trực quan hóa dữ liệu bao gồm hiểu rõ hình ảnh nào phù hợp nhất với dữ liệu mà tổ chức đang sử dụng và các điểm chính mà tổ chức hy vọng đạt được, giữ cho hình ảnh rõ ràng và đơn giản

nhất có thể, đồng thời cung cấp giải thích và nội dung phù hợp để giúp đảm bảo rằng khán giả hiểu những gì họ đang xem.

7.2.7.5. Quản lý dữ liệu

Việc quản lý dữ liệu liên tục được tiến hành song song với những gì đã được đề cập trước đó. Một tổ chức sử dụng phân tích kinh doanh phải tạo ra một chiến lược toàn diện để duy trì dữ liệu sạch của mình, đặc biệt khi tổ chức này kết hợp các nguồn dữ liệu mới.

7.3. Học máy

Định nghĩa: Học máy (ML) là một nhánh của trí tuệ nhân tạo và khoa học máy tính, tập trung vào việc sử dụng dữ liệu và thuật toán để cho phép AI bắt chước cách con người học, dần dần cải thiện độ chính xác của nó.

7.3.1. Hoạt động của học máy

UC Berkeley chia hệ thống học tập của thuật toán học máy thành ba phần chính.

1. *Quy trình ra quyết định:* Nói chung, thuật toán học máy được sử dụng để đưa ra dự đoán hoặc phân loại. Dựa trên một số dữ liệu đầu vào, có thể được gắn nhãn hoặc không được gắn nhãn, thuật toán sẽ đưa ra ước tính về một mẫu trong dữ liệu.
2. *Hàm lỗi*: Hàm lỗi đánh giá dự đoán của mô hình. Nếu có các ví dụ đã biết, hàm lỗi có thể so sánh để đánh giá độ chính xác của mô hình.
3. *Quy trình tối ưu hóa mô hình:* Nếu mô hình có thể phù hợp hơn với các điểm dữ liệu trong tập huấn luyện thì các trọng số sẽ được điều chỉnh để giảm sự khác biệt giữa ví dụ đã biết và ước tính mô hình. Thuật toán sẽ lặp lại quá trình "đánh giá và tối ưu hóa" lặp đi lặp lại này, cập nhật trọng số một cách tự động cho đến khi đạt đến ngưỡng chính xác.

7.3.2. Học máy, học sâu và mạng lưới thần kinh

Vì học sâu và học máy có xu hướng được sử dụng thay thế cho nhau nên cần lưu ý các sắc thái giữa hai điều này. Học máy, học sâu và mạng lưới thần kinh đều là các lĩnh vực con của trí tuệ nhân tạo. Tuy nhiên, mạng lưới thần kinh thực sự là một lĩnh vực con của học máy và học sâu là một lĩnh vực con của mạng lưới thần kinh.

Sự khác biệt giữa deep learning và machine learning nằm ở cách mỗi thuật toán học. Học máy "sâu" có thể sử dụng các tập dữ liệu được gắn nhãn, còn được gọi là học có giám sát, để thông báo cho thuật toán của nó, nhưng nó không nhất thiết yêu cầu một tập dữ liệu được gắn nhãn. Quá trình học sâu có thể sử dụng dữ liệu phi cấu trúc ở dạng thô (ví dụ: văn bản hoặc hình ảnh) và nó có thể tự động xác định tập hợp các tính năng giúp phân biệt các loại dữ liệu khác nhau. Điều này giúp loại bỏ một số sự can thiệp cần thiết của con người và cho phép sử dụng lượng lớn dữ liệu. Có thể coi học sâu là "học máy có khả năng mở rộng" như Lex Fridman lưu ý trong bài giảng MIT này (liên kết nằm bên ngoài ibm.com).

Học máy cổ điển hoặc "không sâu" phụ thuộc nhiều hơn vào sự can thiệp của con người để học. Các chuyên gia về con người xác định tập hợp các tính năng để hiểu sự khác biệt giữa các dữ liệu đầu vào, thường yêu cầu nhiều dữ liệu có cấu trúc hơn để tìm hiểu.

Mạng thần kinh hoặc mạng thần kinh nhân tạo (ANN), bao gồm các lớp nút, chứa một lớp đầu vào, một hoặc nhiều lớp ẩn và một lớp đầu ra. Mỗi nút hoặc nơron nhân tạo kết nối với một nút khác và có trọng số cũng như ngưỡng liên quan. Nếu đầu ra của bất kỳ nút riêng lẻ nào cao hơn giá trị ngưỡng được chỉ định thì nút đó sẽ được kích hoạt, gửi dữ liệu đến lớp tiếp theo của mạng. Mặt khác, nút đó sẽ không truyền dữ liệu đến lớp tiếp theo của mạng. Từ “deep” trong deep learning chỉ đề cập đến số lượng lớp trong mạng lưới thần kinh. Mạng nơ-ron bao gồm nhiều hơn ba lớp—bao gồm đầu vào và đầu ra—có thể được coi là thuật toán học sâu hoặc mạng nơ-ron sâu. Mạng nơ-ron chỉ có ba lớp chỉ là mạng nơ-ron cơ bản.

Mạng lưới thần kinh và học sâu được ghi nhận là có khả năng thúc đẩy tiến bộ trong các lĩnh vực như thị giác máy tính, xử lý ngôn ngữ tự nhiên và nhận dạng giọng nói.

7.3.3. Phương pháp học máy

Các mô hình học máy được chia thành ba loại chính.

7.3.3.1. Học máy được giám sát

Học có giám sát, còn được gọi là học máy có giám sát, được xác định bằng cách sử dụng các bộ dữ liệu được gắn nhãn để đào tạo các thuật toán nhằm phân loại dữ liệu hoặc dự đoán kết quả một cách chính xác. Khi dữ liệu đầu vào được đưa vào mô hình, mô hình sẽ điều chỉnh trọng số của nó cho

đến khi nó được lắp phù hợp. Điều này xảy ra như một phần của quá trình xác nhận chéo để đảm bảo rằng mô hình tránh được việc trang bị quá mức hoặc thiếu trang bị. Học tập có giám sát giúp các tổ chức giải quyết nhiều vấn đề trong thế giới thực trên quy mô lớn, chẳng hạn như phân loại thư rác vào một thư mục riêng biệt với hộp thư đến. Một số phương pháp được sử dụng trong học tập có giám sát bao gồm mạng lưới thần kinh, vịnh ngây thơ, hồi quy tuyến tính, hồi quy logistic, rừng ngẫu nhiên và máy vectơ hỗ trợ (SVM).

7.3.3.2. Học máy không giám sát

Học không giám sát, còn được gọi là học máy không giám sát, sử dụng thuật toán học máy để phân tích và phân cụm các tập dữ liệu không được gắn nhãn (các tập hợp con được gọi là cụm). Các thuật toán này khám phá các mẫu hoặc nhóm dữ liệu ẩn mà không cần sự can thiệp của con người. Khả năng khám phá những điểm tương đồng và khác biệt về thông tin của phương pháp này khiến nó trở nên lý tưởng cho việc phân tích dữ liệu khám phá, chiến lược bán chéo, phân khúc khách hàng cũng như nhận dạng hình ảnh và mẫu. Nó cũng được sử dụng để giảm số lượng tính năng trong mô hình thông qua quá trình giảm kích thước. Phân tích thành phần chính (PCA) và phân tích giá trị số ít (SVD) là hai phương pháp phổ biến cho việc này. Các thuật toán khác được sử dụng trong học tập không giám sát bao gồm mạng lưới thần kinh, phân cụm k-mean và phương pháp phân cụm xác suất.

7.3.3.3. Học bán giám sát

Học bán giám sát cung cấp một phương tiện hài hòa giữa học có giám sát và không giám sát. Trong quá trình đào tạo, nó sử dụng tập dữ liệu được gắn nhãn nhỏ hơn để hướng dẫn phân loại và trích xuất tính năng từ tập dữ liệu lớn hơn, không được gắn nhãn. Học bán giám sát có thể giải quyết vấn đề không có đủ dữ liệu được dán nhãn cho thuật toán học có giám sát. Nó cũng hữu ích nếu việc dán nhãn đủ dữ liệu quá tốn kém.

7.3.4. Học máy tăng cường

Học máy tăng cường là một mô hình học máy tương tự như học có giám sát, nhưng thuật toán không được đào tạo bằng dữ liệu mẫu. Mô hình này học hỏi bằng cách sử dụng phương pháp thử và sai. Một chuỗi các kết quả thành công sẽ được củng cố để phát triển khuyến nghị hoặc chính sách tốt nhất cho một vấn đề nhất định.

Hệ thống IBM Watson® đã giành chiến thắng trong cuộc thi Jeopardy! thách thức năm 2011 là một ví dụ điển hình. Hệ thống sử dụng phương pháp học tăng cường để tìm hiểu khi nào nên thử trả lời (hoặc câu hỏi), ô nào cần chọn trên bảng và số tiền đặt cược—đặc biệt là đối với các cược gấp đôi hàng ngày.

7.3.5. Các thuật toán học máy phổ biến

Một số thuật toán học máy thường được sử dụng. Bao gồm các:

- *Mạng thần kinh*: Mạng lưới thần kinh mô phỏng cách thức hoạt động của bộ não con người, với số lượng lớn các nút xử lý được liên kết với nhau. Mạng lưới thần kinh rất giỏi trong việc nhận dạng các mẫu và đóng vai trò quan trọng trong các ứng dụng bao gồm dịch ngôn ngữ tự nhiên, nhận dạng hình ảnh, nhận dạng giọng nói và tạo hình ảnh.
- *Hồi quy tuyến tính*: Thuật toán này được sử dụng để dự đoán các giá trị số, dựa trên mối quan hệ tuyến tính giữa các giá trị khác nhau. Ví dụ, kỹ thuật này có thể được sử dụng để dự đoán giá nhà dựa trên dữ liệu lịch sử của khu vực.
- *Hồi quy logistic*: Thuật toán học có giám sát này đưa ra dự đoán cho các biến phản hồi được phân loại, chẳng hạn như câu trả lời "có/không" cho các câu hỏi. Nó có thể được sử dụng cho các ứng dụng như phân loại thư rác và kiểm soát chất lượng trên dây chuyền sản xuất.
- *Phân cụm*: Sử dụng phương pháp học không giám sát, các thuật toán phân cụm có thể xác định các mẫu trong dữ liệu để có thể nhóm lại. Máy tính có thể giúp các nhà khoa học dữ liệu bằng cách xác định sự khác biệt giữa các mục dữ liệu mà con người đã bỏ qua.
- *Cây quyết định*: Cây quyết định có thể được sử dụng cho cả việc dự đoán các giá trị số (hồi quy) và phân loại dữ liệu thành các danh mục. Cây quyết định sử dụng chuỗi phân nhánh của các quyết định được liên kết có thể được biểu diễn bằng sơ đồ cây. Một trong những ưu điểm của cây quyết định là chúng dễ dàng xác thực và kiểm tra, không giống như hộp đen của mạng lưới thần kinh.

- *Rừng ngẫu nhiên*: Trong rừng ngẫu nhiên, thuật toán học máy dự đoán một giá trị hoặc danh mục bằng cách kết hợp các kết quả từ một số cây quyết định.

7.3.6. Ưu điểm và nhược điểm của thuật toán học máy

Tùy thuộc vào ngân sách, nhu cầu về tốc độ và độ chính xác cần thiết, mỗi loại thuật toán (được giám sát, không giám sát, bán giám sát hoặc tăng cường) có những ưu điểm và nhược điểm riêng.Ví dụ: thuật toán cây quyết định được sử dụng cho cả việc dự đoán các giá trị số (bài toán hồi quy) và phân loại dữ liệu thành các danh mục. Cây quyết định sử dụng chuỗi phân nhánh của các quyết định được liên kết có thể được biểu diễn bằng sơ đồ cây. Ưu điểm chính của cây quyết định là chúng dễ xác thực và kiểm tra hơn mạng nơ-ron. Tin xấu là chúng có thể không ổn định hơn các yếu tố dự báo quyết định khác.

Nhìn chung, học máy có nhiều lợi thế mà doanh nghiệp có thể tận dụng để đạt được hiệu quả mới. Chúng bao gồm việc học máy xác định các mô hình và xu hướng trong khối lượng dữ liệu khổng lồ mà con người có thể không phát hiện ra. Và phân tích này đòi hỏi ít sự can thiệp của con người: chỉ cần cung cấp dữ liệu quan tâm và để hệ thống máy học tập hợp và tinh chỉnh các thuật toán của riêng nó—điều này sẽ liên tục cải thiện với nhiều dữ liệu đầu vào hơn theo thời gian. Khách hàng và người dùng có thể tận hưởng trải nghiệm được cá nhân hóa hơn khi mô hình học hỏi được nhiều hơn qua mỗi trải nghiệm với người đó.

Mặt khác, học máy yêu cầu tập dữ liệu đào tạo lớn, chính xác và không thiên vị. GIGO[1] là hệ số hoạt động: rác vào/rác ra. Thu thập đủ dữ liệu và có một hệ thống đủ mạnh để chạy nó cũng có thể làm tiêu hao tài nguyên. Học máy cũng có thể dễ bị lỗi, tùy thuộc vào đầu vào. Với mẫu quá nhỏ, hệ thống có thể tạo ra một thuật toán logic hoàn toàn sai hoặc gây hiểu lầm. Để tránh lãng phí ngân sách hoặc làm mất lòng khách hàng, các tổ chức chỉ nên hành động dựa trên các câu trả lời khi có độ tin cậy cao về kết quả đầu ra.

7.3.7. Các trường hợp sử dụng máy học trong thế giới thực

Đây chỉ là một vài ví dụ về học máy mà người ta có thể gặp hàng ngày:

[1] Garbage in, garbage out GIGO

- Nhận dạng giọng nói: Nó còn được gọi là nhận dạng giọng nói tự động (ASR[1]), nhận dạng giọng nói máy tính hoặc chuyển giọng nói thành văn bản và là khả năng sử dụng xử lý ngôn ngữ tự nhiên (NLP[2]) để dịch lời nói của con người sang định dạng viết. Nhiều thiết bị di động kết hợp nhận dạng giọng nói vào hệ thống của họ để tiến hành tìm kiếm bằng giọng nói—ví dụ: Siri—hoặc cải thiện khả năng truy cập để nhắn tin.
- Dịch vụ khách hàng: Chatbot trực tuyến đang thay thế các tác nhân con người trong hành trình của khách hàng, thay đổi cách chúng ta nghĩ về sự tương tác của khách hàng trên các trang web và nền tảng truyền thông xã hội. Chatbots trả lời các câu hỏi thường gặp (FAQ[3]) về các chủ đề như vận chuyển hoặc cung cấp lời khuyên được cá nhân hóa, bán chéo sản phẩm hoặc đề xuất kích thước cho người dùng. Ví dụ bao gồm các đại lý ảo trên các trang thương mại điện tử; bot nhắn tin, sử dụng Slack và Facebook Messenger; và các công việc thường được thực hiện bởi trợ lý ảo và trợ lý giọng nói.
- Thị giác máy tính: Công nghệ AI này cho phép máy tính lấy được thông tin có ý nghĩa từ hình ảnh kỹ thuật số, video và các đầu vào trực quan khác, sau đó thực hiện hành động thích hợp. Được hỗ trợ bởi mạng lưới thần kinh tích chập, thị giác máy tính có các ứng dụng trong việc gắn thẻ ảnh trên mạng xã hội, chụp ảnh X quang trong chăm sóc sức khỏe và xe tự lái trong ngành công nghiệp ô tô.
- Công cụ đề xuất: Sử dụng dữ liệu hành vi tiêu dùng trong quá khứ, thuật toán AI có thể giúp khám phá các xu hướng dữ liệu có thể được sử dụng để phát triển các chiến lược bán chéo hiệu quả hơn. Công cụ đề xuất được các nhà bán lẻ trực tuyến sử dụng để đưa ra đề xuất sản phẩm phù hợp cho khách hàng trong quá trình thanh toán.

[1] Automatic speech recognition ASR
[2] Natural language processing NLP
[3] Frequently asked questions: FAQ

- Tự động hóa quy trình bằng robot (RPA[1]): Còn được gọi là robot phần mềm, RPA sử dụng các công nghệ tự động hóa thông minh để thực hiện các tác vụ thủ công lặp đi lặp lại.

7.3.8. Những thách thức của học máy

Khi công nghệ máy học phát triển, nó chắc chắn đã giúp cuộc sống của chúng ta trở nên dễ dàng hơn. Tuy nhiên, việc triển khai học máy trong doanh nghiệp cũng làm nảy sinh một số lo ngại về mặt đạo đức đối với công nghệ AI. Một số trong số này bao gồm:

8.1. Điểm kỳ dị về công nghệ

Trong khi chủ đề này thu hút được nhiều sự chú ý của công chúng thì nhiều nhà nghiên cứu lại không mấy quan tâm đến ý tưởng AI sẽ vượt qua trí thông minh của con người trong tương lai gần. Điểm kỳ dị về công nghệ còn được gọi là AI mạnh hay siêu trí tuệ. Triết gia Nick Bostrum định nghĩa siêu trí tuệ là "bất kỳ trí tuệ nào vượt trội hơn hẳn những bộ não tốt nhất của con người trong hầu hết mọi lĩnh vực, bao gồm tính sáng tạo khoa học, trí tuệ tổng quát và kỹ năng xã hội". Mặc dù thực tế là siêu trí tuệ chưa xuất hiện trong xã hội nhưng ý tưởng về nó đặt ra một số câu hỏi thú vị khi chúng ta xem xét việc sử dụng các hệ thống tự hành, như ô tô tự lái. Thật phi thực tế khi nghĩ rằng một chiếc ô tô không người lái sẽ không bao giờ gặp tai nạn, nhưng ai là người chịu trách nhiệm và chịu trách nhiệm trong những trường hợp đó? Chúng ta có nên tiếp tục phát triển phương tiện tự hành hay chỉ giới hạn công nghệ này ở những phương tiện bán tự động giúp mọi người lái xe an toàn? Ban giám khảo vẫn chưa đưa ra quan điểm về vấn đề này, nhưng đây là những kiểu tranh luận về đạo đức đang diễn ra khi công nghệ AI mới, sáng tạo phát triển.

8.2. Tác động của AI đến việc làm

Trong khi nhiều nhận thức của công chúng về trí tuệ nhân tạo xoay quanh tình trạng mất việc làm, mối lo ngại này có lẽ nên được điều chỉnh lại. Với mỗi công nghệ mới, mang tính đột phá, chúng tôi thấy rằng nhu cầu của thị trường đối với các vai trò công việc cụ thể đều thay đổi. Ví dụ, khi chúng ta nhìn vào ngành công nghiệp ô tô, nhiều nhà sản xuất, như GM, đang chuyển sang tập trung vào sản xuất xe điện để phù hợp với các sáng kiến

[1] Robotic process automation RPA

xanh. Ngành năng lượng sẽ không biến mất, nhưng nguồn năng lượng đang chuyển từ tiết kiệm nhiên liệu sang tiết kiệm điện.

Tương tự, trí tuệ nhân tạo sẽ chuyển nhu cầu việc làm sang các lĩnh vực khác. Sẽ cần có những cá nhân giúp quản lý hệ thống AI. Vẫn cần có người để giải quyết các vấn đề phức tạp hơn trong các ngành có nhiều khả năng bị ảnh hưởng nhất bởi sự thay đổi nhu cầu công việc, chẳng hạn như dịch vụ khách hàng. Thách thức lớn nhất với trí tuệ nhân tạo và tác động của nó đối với thị trường việc làm sẽ là giúp mọi người chuyển đổi sang các vai trò mới đang có nhu cầu.

8.3. Sự riêng tư

Quyền riêng tư có xu hướng được thảo luận trong bối cảnh bảo mật dữ liệu, bảo vệ dữ liệu và bảo mật dữ liệu. Những lo ngại này đã cho phép các nhà hoạch định chính sách đạt được nhiều bước tiến hơn trong những năm gần đây. Ví dụ: vào năm 2016, luật GDPR đã được ban hành để bảo vệ dữ liệu cá nhân của những người ở Liên minh Châu Âu và Khu vực Kinh tế Châu Âu, giúp các cá nhân có nhiều quyền kiểm soát dữ liệu của mình hơn. Tại Hoa Kỳ, từng bang đang xây dựng các chính sách, chẳng hạn như Đạo luật về quyền riêng tư của người tiêu dùng California (CCPA), được ban hành vào năm 2018 và yêu cầu các doanh nghiệp thông báo cho người tiêu dùng về việc thu thập dữ liệu của họ. Pháp luật như thế này đã buộc các công ty phải suy nghĩ lại về cách họ lưu trữ và sử dụng thông tin nhận dạng cá nhân (PII). Do đó, đầu tư vào bảo mật ngày càng trở thành ưu tiên hàng đầu của các doanh nghiệp khi họ tìm cách loại bỏ mọi lỗ hổng và cơ hội để giám sát, hack và tấn công mạng.

8.4. Sự thiên vị và phân biệt đối xử

Các trường hợp thiên vị và phân biệt đối xử trên một số hệ thống máy học đã đặt ra nhiều câu hỏi về đạo đức liên quan đến việc sử dụng trí tuệ nhân tạo. Làm cách nào chúng ta có thể bảo vệ khỏi sự thiên vị và phân biệt đối xử khi bản thân dữ liệu đào tạo có thể được tạo ra bởi các quy trình thiên vị của con người? Trong khi các công ty thường có ý định tốt cho nỗ lực tự động hóa của họ, thì Reuters (liên kết nằm bên ngoài ibm.com) nhấn mạnh một số hậu quả không lường trước được của việc kết hợp AI vào hoạt động tuyển dụng. Trong nỗ lực tự động hóa và đơn giản hóa quy trình, Amazon đã vô tình phân biệt đối xử với các ứng viên theo giới tính cho các vị trí kỹ thuật

và cuối cùng công ty đã phải hủy bỏ dự án. Harvard Business Review (liên kết nằm bên ngoài ibm.com) đã đặt ra các câu hỏi rõ ràng khác về việc sử dụng AI trong thực tiễn tuyển dụng, chẳng hạn như dữ liệu nào có thể sử dụng khi đánh giá ứng viên cho một vai trò.

Thành kiến và phân biệt đối xử cũng không chỉ giới hạn ở chức năng nhân sự; chúng có thể được tìm thấy trong một số ứng dụng từ phần mềm nhận dạng khuôn mặt đến các thuật toán truyền thông xã hội.

Khi các doanh nghiệp nhận thức rõ hơn về những rủi ro với AI, họ cũng trở nên tích cực hơn trong cuộc thảo luận về đạo đức và giá trị của AI. Ví dụ, IBM đã ngừng cung cấp các sản phẩm phân tích và nhận dạng khuôn mặt cho mục đích chung. Giám đốc điều hành IBM Arvind Krishna đã viết: "IBM kiên quyết phản đối và sẽ không tha thứ cho việc sử dụng bất kỳ công nghệ nào, bao gồm cả công nghệ nhận dạng khuôn mặt do các nhà cung cấp khác cung cấp, để giám sát hàng loạt, lập hồ sơ chủng tộc, vi phạm các quyền và tự do cơ bản của con người hoặc bất kỳ mục đích nào không nhất quán." với các giá trị và Nguyên tắc Tin cậy và Minh bạch của chúng tôi."

7.3.8,5. Trách nhiệm giải trình

Vì không có luật quan trọng để điều chỉnh các hoạt động AI nên không có cơ chế thực thi thực sự nào để đảm bảo rằng AI có đạo đức được thực hành. Những khuyến khích hiện tại để các công ty tuân thủ đạo đức là những tác động tiêu cực của một hệ thống AI phi đạo đức ở điểm mấu chốt. Để lấp đầy khoảng trống, các khuôn khổ đạo đức đã xuất hiện như một phần của sự hợp tác giữa các nhà đạo đức và nhà nghiên cứu nhằm quản lý việc xây dựng và phân phối các mô hình AI trong xã hội. Tuy nhiên, hiện tại, những điều này chỉ có tác dụng hướng dẫn. Một số nghiên cứu (liên kết nằm bên ngoài ibm.com) cho thấy rằng sự kết hợp giữa trách nhiệm được phân bổ và việc thiếu tầm nhìn xa về những hậu quả tiềm ẩn không có lợi cho việc ngăn ngừa tác hại cho xã hội.

7.3.9. Cách chọn nền tảng AI phù hợp cho học máy

Việc lựa chọn một nền tảng có thể là một quá trình đầy thử thách vì hệ thống sai có thể làm tăng chi phí hoặc hạn chế việc sử dụng các công cụ hoặc công nghệ có giá trị khác. Khi xem xét nhiều nhà cung cấp để chọn nền tảng AI, thường có xu hướng nghĩ rằng nhiều tính năng hơn = hệ thống tốt hơn. Có thể vậy, nhưng người đánh giá nên bắt đầu bằng cách suy nghĩ xem

nền tảng AI sẽ làm gì cho tổ chức của họ. Những khả năng học máy nào cần được cung cấp và những tính năng nào quan trọng để đạt được chúng? Một tính năng bị thiếu có thể làm mất tính hữu dụng của toàn bộ hệ thống. Dưới đây là một số tính năng cần xem xét.

Khả năng của các hoạt động học máy. Hệ thống có:

- một giao diện thống nhất để dễ quản lý?
- các công cụ học máy tự động để tạo mô hình nhanh hơn với chức năng mã ngắn và không mã?
- tối ưu hóa quyết định để hợp lý hóa việc lựa chọn và triển khai các mô hình tối ưu hóa?
- mô hình hóa trực quan để kết hợp khoa học dữ liệu trực quan với các thư viện nguồn mở và giao diện dựa trên sổ ghi chép trên một studio dữ liệu và AI thống nhất?
- phát triển tự động cho người mới bắt đầu để bắt đầu nhanh chóng và các nhà khoa học dữ liệu tiên tiến hơn để thử nghiệm?
- Trình tạo dữ liệu tổng hợp để thay thế hoặc bổ sung cho dữ liệu trong thế giới thực khi dữ liệu trong thế giới thực không có sẵn?

Khả năng AI sáng tạo. Hệ thống có:

- trình tạo nội dung có thể tạo văn bản, hình ảnh và nội dung khác dựa trên dữ liệu đã được đào tạo?
- phân loại tự động để đọc và phân loại văn bản đầu vào, chẳng hạn như đánh giá và phân loại khiếu nại của khách hàng hoặc xem xét cảm nhận phản hồi của khách hàng?
- một trình tạo bản tóm tắt có thể chuyển đổi văn bản dày đặc thành bản tóm tắt chất lượng cao, nắm bắt các điểm chính từ báo cáo tài chính và tạo bản ghi cuộc họp?
- khả năng trích xuất dữ liệu để sắp xếp các chi tiết phức tạp và nhanh chóng lấy thông tin cần thiết từ các tài liệu lớn?

7.4. Thực hiện hệ thống thông minh trong dự án học máy

7.4.1. Xây dựng mô hình với TensorFlow

Từ trợ lý ảo đến ô tô tự lái, các công ty công nghệ đang chạy đua tung ra sản phẩm và nâng cao trải nghiệm người dùng bằng cách khám phá các khả năng của Trí tuệ nhân tạo (AI). Rõ ràng là từ báo cáo Market Research Future Indeed cho thấy thị trường việc làm học máy được dự đoán sẽ trị giá gần 31 tỷ USD vào năm 2024. Học máy tạo ra các thuật toán cho phép máy học và áp dụng trí thông minh mà không cần được chỉ đạo, và TensorFlow là một thư viện nguồn mở được sử dụng để xây dựng các mô hình học máy.

7.4.1.1. Về học sâu

Học sâu là một tập hợp con của học máy. Có một số chuyên môn nhất định mà chúng tôi thực hiện học máy và đó là lý do tại sao nó được gọi là học sâu. Ví dụ, deep learning sử dụng mạng lưới thần kinh, giống như một mô phỏng của bộ não con người. Học sâu cũng liên quan đến việc phân tích một lượng lớn dữ liệu phi cấu trúc, không giống như học máy truyền thống thường sử dụng dữ liệu có cấu trúc. Dữ liệu phi cấu trúc này có thể được cung cấp dưới dạng hình ảnh, video, âm thanh, văn bản, v.v.

Thuật ngữ 'sâu' xuất phát từ thực tế là mạng lưới thần kinh có thể có nhiều lớp ẩn.

7.4.1.2. Thư viện phổ biến cho học sâu

Thư viện học máy Python đã trở thành ngôn ngữ thực hiện các thuật toán học máy. Cần sử dụng được Python. Dưới đây là các thư viện học máy Python năm 2022.

- *TensorFlow*. TensorFlow là một thư viện máy tính số mã nguồn mở cho machine learning dựa trên mạng nơ-ron nhân tạo. Nó được tạo ra bởi nhóm nghiên cứu Google Brain vào năm 2015 để sử dụng nội bộ trong các sản phẩm của Google.
- *PyTorch.* PyTorch là một trong những thư viện machine learning lớn nhất được thiết kế và phát triển bởi nhóm nghiên cứu AI của Facebook. Nó được sử dụng để xử lý ngôn ngữ tự nhiên, thị giác máy tính và các loại tác vụ tương tự khác.

- *Keras.* Keras là một nền tảng thử nghiệm nhanh với các mạng nơ-ron sâu nhưng nó đã sớm có được một thư viện Python ML độc lập. Nó có giao diện thân thiện với người dùng và có hỗ trợ đa phụ trợ.
- *Orage3*. Orage3 là một gói phần mềm bao gồm các công cụ cho machine learining, khai thác dữ liệu và trực quan hóa dữ liệu. Nó được phát triển vào năm 1996, các nhà khoa học tại Đại học Ljubljana đã tạo ra nó bằng C++.
- *NumPy.* Ban đầu, Python không được phát triển như một công cụ cho tính toán số. Sự ra đời của NumPy là chìa khóa để mở rộng khả năng của Python dưới dạng các hàm toán học, dựa trên đó các giải pháp machine learning sẽ được xây dựng. Sử dụng thư viện này có lợi vì khả năng tính toán mạnh mẽ, cộng đồng lập trình lớn và hiệu suất cao.
- *SciPy.* Cùng với NumPy, thư viện này là một công cụ cốt lõi để hoàn thành các phép tính toán học, kỹ thuật và khoa học. Các lý do chính khiến các chuyên gia Python đánh giá cao SciPy là thư viện dễ sử dụng, khả năng tính toán nhanh và tính toán được cải thiện. SciPy được xây dựng dựa trên NumPy và có thể hoạt động trên các mảng của nó, đảm bảo chất lượng cao hơn và thực thi nhanh hơn các hoạt động tính toán.
- *Scikit-Learn.* Scikit-learning đầu tiên được tạo ra như một phần mở rộng của bên thứ ba cho thư viện SciPy. Nó là một trong những thư viện hàng đầu trên GitHub.
- *Pandas*. Pandas là một thư viện Python cấp thấp được xây dựng dựa trên NumPy. Pandas có khung dữ liệu mạnh mẽ và xử lý dữ liệu linh hoạt.
- *Matplotlib.* Sự thống nhất của NumPy, Matplotlib và SciPy được cho là sẽ thay thế nhu cầu sử dụng ngôn ngữ thống kê MATLAB. Các gói Python được miễn phí và linh hoạt hơn có thể là lựa chọn của nhiều nhà khoa học dữ liệu.
- *Theano.* Vào năm 2007, Viện thuật toán học Montreal[1] được thành lập bởi Theano để đánh giá và thao tác các biểu thức toán học khác nhau.

[1] Montreal Institute of Learning Algorithms

7.4.1.3. Về TensorFlow

TensorFlow là một thư viện mã nguồn mở được nhóm Google Brain phát triển vào năm 2012. Python cho đến nay là ngôn ngữ phổ biến nhất mà TensorFlow sử dụng. Có thể nhập thư viện TensorFlow vào môi trường Python của mình và thực hiện phát triển học tập chuyên sâu.

Có một cách chắc chắn để chương trình được thực thi. Trước tiên, tạo các nút để xử lý dữ liệu ở dạng biểu đồ. Dữ liệu được lưu trữ dưới dạng tensor và dữ liệu tensor truyền đến các nút khác nhau.

7.4.2. Lí do nên sử dụng TensorFlow

7.4.2.1. Giới thiệu

Một trong những ưu điểm tốt nhất của TensorFlow là nó giúp việc phát triển mã trở nên dễ dàng. Các API sẵn có giúp người dùng không phải viết lại một số mã mà lẽ ra sẽ tốn thời gian. TensorFlow tăng tốc quá trình đào tạo mô hình. Ngoài ra, khả năng xảy ra lỗi trong chương trình cũng giảm đi, thường từ 55 đến 85%.

Khía cạnh quan trọng khác là TensorFlow có khả năng mở rộng cao. Có thể viết mã của mình rồi làm cho mã chạy trên CPU, GPU hoặc trên một cụm hệ thống này cho mục đích đào tạo.

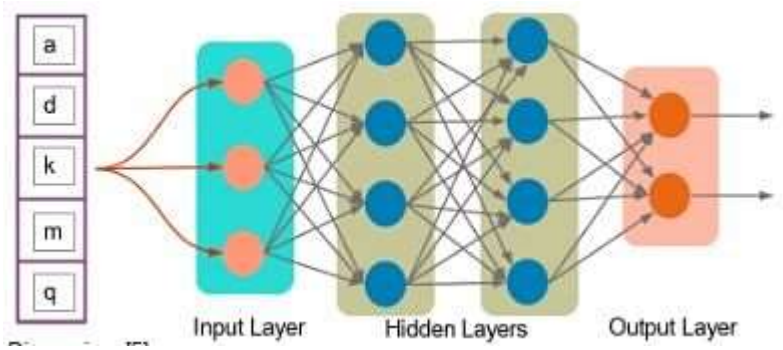
Nói chung, việc huấn luyện mô hình là nơi thực hiện phần lớn công việc tính toán. Ngoài ra, quá trình đào tạo được lặp lại nhiều lần để giải quyết mọi vấn đề có thể phát sinh. Quá trình này dẫn đến việc tiêu thụ nhiều năng lượng hơn và do đó, cần một máy tính phân tán. Nếu cần xử lý lượng lớn dữ liệu, TensorFlow sẽ giúp việc này trở nên dễ dàng bằng cách chạy mã theo cách phân tán.

GPU, hay đơn vị xử lý đồ họa, đã trở nên rất phổ biến. Nvidia là một trong những công ty dẫn đầu trong lĩnh vực này. Nó thực hiện tốt các phép tính toán học, chẳng hạn như phép nhân ma trận và đóng một vai trò quan trọng trong việc học sâu. TensorFlow cũng tích hợp với API C++ và Python, giúp quá trình phát triển nhanh hơn nhiều.

Trước khi xem hướng dẫn về TensorFlow này, nên biết TensorFlow thực sự là gì.

7.4.2.2. Khái niệm về Tenxơ

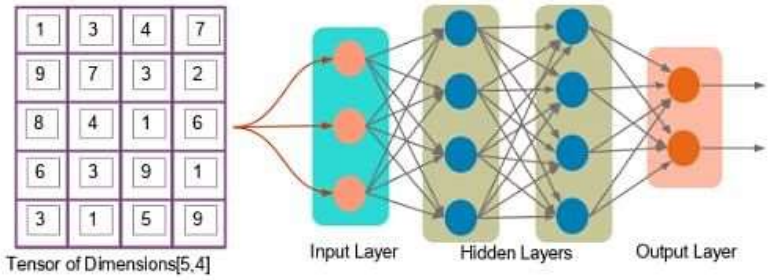
Tensor là một đối tượng toán học được biểu diễn dưới dạng mảng có kích thước cao hơn. Những mảng dữ liệu với kích thước và thứ hạng khác nhau này được cung cấp làm đầu vào cho mạng lưới thần kinh. Đây là những tensor.



Hình. Tensor như một mảng

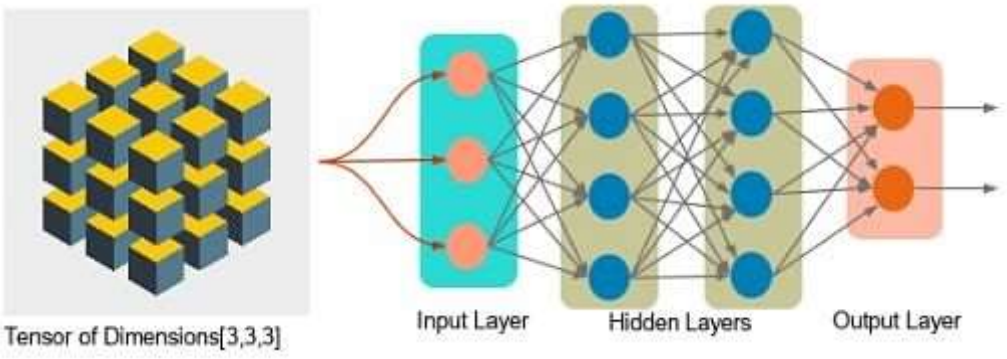
Có thể có các mảng hoặc vectơ là một chiều hoặc ma trận là hai chiều. Nhưng tensor có thể nhiều hơn ba, bốn hoặc năm chiều. Do đó, nó giúp giữ dữ liệu rất chặt chẽ ở một nơi và sau đó thực hiện tất cả các phân tích xung quanh đó.

Chúng ta hãy xem một ví dụ về một tenxơ có kích thước [5,4] (hai chiều).



Hình. Tensor hai chiều

Tiếp theo, có thể thấy một tensor có chiều [3,3,3] (ba chiều).



Hình. Các tensor

Chúng ta hãy thảo luận về Xếp hạng Tensor trong phần tiếp theo.

7.4.2.3. Hạng tenxơ

Thứ hạng của tenxơ không là gì ngoài kích thước của tenxơ. Nó bắt đầu bằng số không. Số 0 là một đại lượng vô hướng không có nhiều mục trong đó. Đó là một giá trị duy nhất.

Ví dụ: s = 10 là một tensor hạng 0 hoặc một đại lượng vô hướng.

V = [10, 11, 12] là một tensor hạng 1 hoặc một vectơ.

M = [[1, 2, 3],[4, 5, 6]] là một tensor hạng 2 hoặc một ma trận.

T = [[[1],[2],[3]],[[4],[5],[6]],[[7],[8],[9]]] là một tensor hạng 3 hoặc một tenxơ.

Có muốn trở thành một kỹ sư AI thành công không? Nếu có, hãy đăng ký Chương trình Thạc sĩ Kỹ sư AI và học AI, Khoa học dữ liệu với Python, Học máy, Học sâu, NLP, được tiếp cận với các phòng thí nghiệm thực tế, các dự án thực hành, v.v.

7.4.2.4. Kiểu dữ liệu Tenxor

Ngoài thứ hạng và hình dạng, tensor còn có kiểu dữ liệu. Sau đây là danh sách các kiểu dữ liệu:

Bảng. Các kiểu dữ liệu

Data Type	Python Type	Description
DF_FLOAT	tf.float32	32 bits floating point
DF_DOUBLE	tf.float64	64 bits floating point
DT_INT8	tf.int8	8 bits signed integer
DT_INT16	tf.int16	16 bits signed integer
DT_INT32	tf.int32	32 bits signed integer
DT_INT64	tf.int64	64 bits signed integer
DT_UINT8	tf.uint8	8 bits unsigned integer
DT_STRING	tf.string	Variable length byte arrays
DT_BOOL	tf.bool	Boolean

Phần tiếp theo của hướng dẫn TensorFlow này thảo luận về cách xây dựng biểu đồ tính toán.

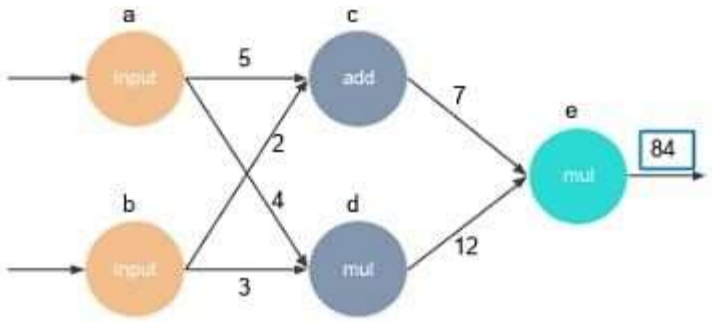
7.4.2.5. Xây dựng đồ thị tính toán

Mọi thứ trong TensorFlow đều dựa trên việc thiết kế biểu đồ tính toán. Biểu đồ có một mạng lưới các nút, với mỗi nút thực hiện phép cộng, nhân hoặc đánh giá một số phương trình nhiều biến.

Mã được viết để xây dựng biểu đồ, tạo phiên và thực thi biểu đồ đó. Một biểu đồ có các nút biểu thị các phép toán và một cạnh biểu thị các tensor. Trong TensorFlow, một phép tính được giải thích bằng biểu đồ luồng dữ liệu.

Mọi thứ đều là một hoạt động. Không chỉ cộng hai biến mà việc tạo một biến cũng là một thao tác. Mỗi khi gán một biến, nó sẽ trở thành một nút. Có thể thực hiện các phép toán, chẳng hạn như phép cộng và phép nhân trên nút đó.

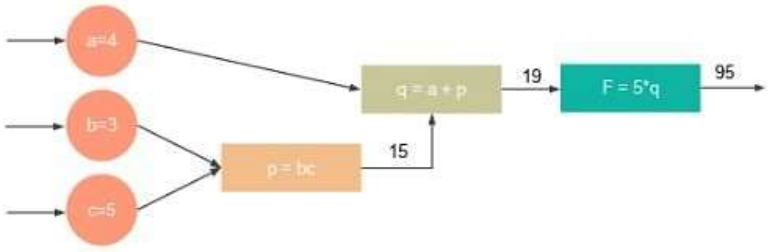
Bắt đầu bằng cách xây dựng các nút này và thực thi chúng ở định dạng đồ họa. Đó là cách cấu trúc chương trình TensorFlow.



Hình. Cấu trúc chương trình

Đây là ví dụ mô tả cách tạo biểu đồ tính toán. Giả sử muốn thực hiện phép tính sau: F(a,b,c) = 5(a+bc)

Ba biến a, b và c chuyển thành ba nút trong biểu đồ, như được hiển thị.



Hình. Ba biến trên biểu đồ

Trong TensorFlow, việc gán các biến này cũng là một thao tác.

Bước 1 là xây dựng biểu đồ bằng cách gán các biến.

Ở đây, các giá trị là:

```
a = 4
b = 3
c = 5
```

Bước 2 của việc xây dựng đồ thị là nhân b và c.

p = b*c

Bước 3 là thêm 'a' vào 'bc.'

q = a + p

Sau đó, chúng ta cần nhiều q và 5.

F = 5*q

Cuối cùng, nhận được kết quả.

Ở đây, chúng tôi có sáu nút. Đầu tiên, xác định từng nút và sau đó tạo một phiên để thực thi nút đó. Đến lượt mình, bước này sẽ quay lại và thực thi từng nút trong số sáu nút để nhận các giá trị đó.

Bây giờ đã biết cách xây dựng biểu đồ tính toán, phần tiếp theo của hướng dẫn TensorFlow này hãy tìm hiểu về các thành phần lập trình.

7.4.2.6. Các phần tử lập trình trong TensorFlow

Không giống như các ngôn ngữ lập trình khác, TensorFlow cho phép gán dữ liệu cho ba thành phần dữ liệu khác nhau:

1. Hằng số

2. Biến

3. Phần giữ chỗ

7.4.2.7. Hằng số

Hằng số là các tham số có giá trị không thay đổi. Chúng ta sử dụng lệnh tf.constant() để xác định một hằng số.

Ví dụ:

```
a = tf.constant(2.0, tf.float32)
b = tf.constant(3.0)
print (a, b)
```

Không thể thay đổi giá trị của hằng số trong quá trình tính toán. Chỉ định dữ liệu, hằng số là tùy chọn.

7.4.2.8. Biến

Các biến cho phép chúng ta thêm các tham số có thể huấn luyện mới vào biểu đồ. Để xác định một biến, chúng ta sử dụng lệnh tf.Variable() và khởi tạo chúng trước khi chạy biểu đồ trong một phiên.

Ví dụ:

```
W = tf.Variable([.3],dtype=tf.float32)
b = tf.Variable([-.3],dtype=tf.float32)
x = tf.placeholder(tf.float32)
```

mô hình tuyến tính = W*x+b

7.4.2.9. Phần giữ chỗ

Trình giữ chỗ cho phép chúng tôi cung cấp dữ liệu cho mô hình TensorFlow từ bên ngoài mô hình. Nó cho phép giá trị được gán sau. Để xác định trình giữ chỗ, chúng tôi sử dụng lệnh tf.placeholder().

Ví dụ:

```
a = tf.placeholder(tf.float32)
b = a*2
```

Với tf.Session() đánh giá:

kết quả = sess.run(b,feed_dict={a:3.0})

7.4.2.10. Kết quả in

Kết quả In có phần giống với một biến nhưng chủ yếu được sử dụng để cung cấp dữ liệu từ bên ngoài. Thông thường, khi thực hiện một bài tập deep learning, không thể lấy tất cả dữ liệu trong một lần và lưu vào bộ nhớ. Điều đó sẽ trở nên không thể quản lý được. Thông thường, sẽ nhận được dữ liệu theo đợt.

Giả sử muốn đào tạo mô hình của mình và có một triệu hình ảnh để thực hiện khóa đào tạo này. Một trong những cách để thực hiện điều này là tạo một biến, tải tất cả hình ảnh và phân tích kết quả. Tuy nhiên, đây có thể không phải là cách tốt nhất vì bộ nhớ có thể bị chậm hoặc có thể xảy ra vấn đề về hiệu suất.

Vấn đề không chỉ là lưu trữ hình ảnh. Cũng cần phải thực hiện đào tạo, đó là một quá trình lặp đi lặp lại. Có thể cần phải tải hình ảnh nhiều lần để huấn luyện mô hình. Đó không chỉ là việc lưu trữ hàng triệu hình ảnh mà còn là quá trình xử lý chiếm nhiều bộ nhớ.

Một cách khác để thực hiện điều này là sử dụng trình giữ chỗ, nơi đọc dữ liệu theo đợt. Và có thể trong số hàng triệu hình ảnh, nhận được hàng nghìn hình ảnh cùng một lúc, xử lý chúng và sau đó nhận được hàng nghìn hình ảnh tiếp theo…

Đó là ý tưởng đằng sau khái niệm phần giữ chỗ; nó chủ yếu được sử dụng để cung cấp dữ liệu cho mô hình. Đọc dữ liệu từ bên ngoài và đưa dữ liệu đó vào biểu đồ bằng tên biến (trong ví dụ: tên biến là Feed_dict).

Khi đang chạy phiên, cần chỉ định cách muốn cung cấp dữ liệu cho mô hình của mình.

7.4.2.11. Phiên

Sau khi tạo biểu đồ, cần thực thi biểu đồ đó bằng cách gọi một phiên hoặc sử dụng phương thức có tên run. Một phiên được chạy để đánh giá các nút, được gọi là thời gian chạy TensorFlow.

Có thể tạo một phiên bằng cách đưa ra lệnh như được hiển thị:

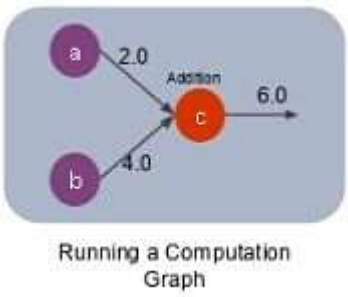
```
sess = tf.Session()
```

Hãy xem xét một ví dụ hiển thị dưới đây:

```
a = tf.constant(5.0)
b = tf.constant(3.0)
c = a*b
# Phiên ra mắt
sess = tf.Session()
# Tính giá trị Tensor c
print(sess.run(c))
```

Có ba nút: a, b và c. Chúng tôi tạo một phiên và chạy nút 'c' vì đây là nơi thực hiện phép toán và thu được kết quả.

Khi chạy nút c, các nút đầu tiên a và b sẽ được tạo, sau đó việc bổ sung sẽ được thực hiện tại nút c. Chúng ta sẽ nhận được kết quả '6' như hình dưới đây:



Hình. Khai thác đồ thị

Phần tiếp theo của hướng dẫn TensorFlow này tập trung vào cách có thể thực hiện hồi quy tuyến tính bằng TensorFlow.

7.4.3. Hồi quy tuyến tính sử dụng TensorFlow

Để xây dựng một mô hình hồi quy tuyến tính bằng TensorFlow, có thể thực hiện các bước sau.

1. Cài đặt TensorFlow: Đảm bảo rằng bạn đã cài đặt TensorFlow. Nếu chưa, bạn có thể cài đặt bằng lệnh sau:
2. Chuẩn bị dữ liệu: Tạo dữ liệu mẫu để huấn luyện mô hình hồi quy tuyến tính.
3. Xây dựng mô hình: Sử dụng API của TensorFlow để xây dựng mô hình hồi quy tuyến tính.
4. Huấn luyện mô hình: Sử dụng dữ liệu để huấn luyện mô hình.
5. Dự đoán: Sử dụng mô hình đã huấn luyện để dự đoán trên dữ liệu mới.

7.4.3.1. Bước 1: Cài đặt TensorFlow

```
pip install tensorflow
```

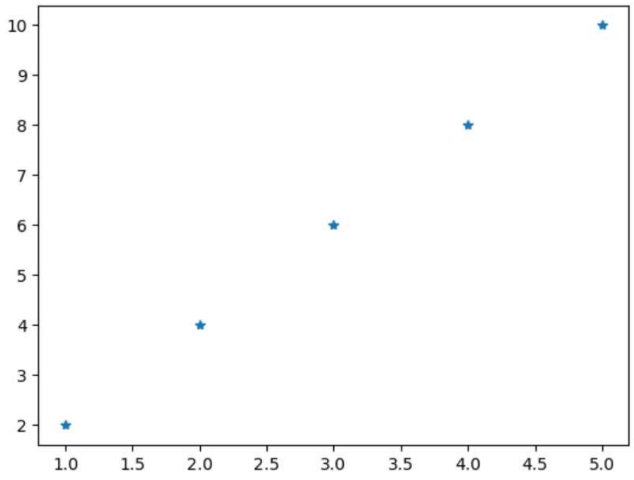
7.4.3.2. Bước 2: Chuẩn bị dữ liệu

Tạo dữ liệu mẫu để huấn luyện mô hình hồi quy tuyến tính.

```
import numpy as np

# Tạo dữ liệu mẫu
X = np.array([1.0, 2.0, 3.0, 4.0, 5.0], dtype=np.float32)
Y = np.array([2.0, 4.0, 6.0, 8.0, 10.0], dtype=np.float32)
```

```
Dữ liệu X
 [1. 2. 3. 4. 5.]
Dữ liệu Y
 [ 2.  4.  6.  8. 10.]
```

Hình. Dữ liệu



Hình. Dạng đồ thị

7.4.3.3. Bước 3: Xây dựng mô hình

Sử dụng TensorFlow để xây dựng mô hình hồi quy tuyến tính.

```
import tensorflow as tf

# Tạo các biến số (weights) của mô hình
W = tf.Variable(initial_value=0.0, dtype=tf.float32,
name='weight')
b = tf.Variable(initial_value=0.0, dtype=tf.float32,
name='bias')

# Hàm dự đoán (prediction) của mô hình
def predict(X):
 return W * X + b
```

7.4.3.4. Bước 4: Huấn luyện mô hình

Sử dụng dữ liệu để huấn luyện mô hình.

```
# Số lần lặp (epochs) và tốc độ học (learning rate)
epochs = 1000
learning_rate = 0.01

# Hàm tính toán mất mát (loss function)
def compute_loss(Y_true, Y_pred):
 return tf.reduce_mean(tf.square(Y_true - Y_pred))

# Tối ưu hóa bằng Gradient Descent
optimizer = tf.optimizers.SGD(learning_rate)

# Quá trình huấn luyện
for epoch in range(epochs):
 with tf.GradientTape() as tape:
 Y_pred = predict(X)
 loss = compute_loss(Y, Y_pred)
 gradients = tape.gradient(loss, [W, b])
 optimizer.apply_gradients(zip(gradients, [W, b]))
 if (epoch + 1) % 100 == 0:
 print(f"Epoch {epoch+1}: Loss = {loss.numpy()}, W =
{W.numpy()}, b = {b.numpy()}")
```

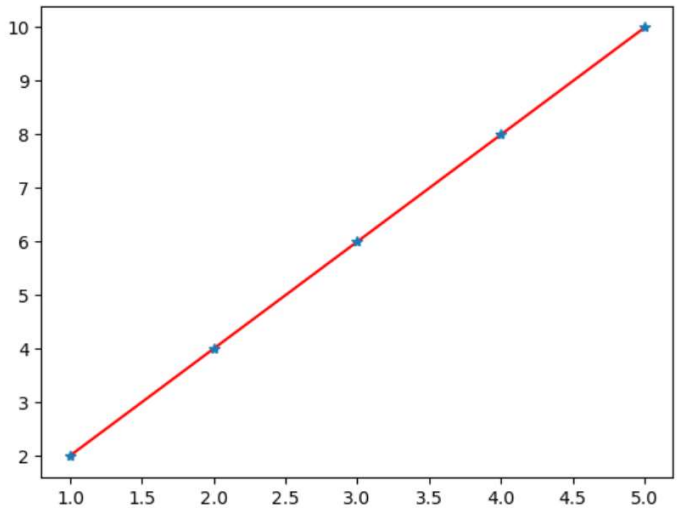
```
Epoch 100: Loss = 0.0246407650411129, W = 1.8984326124191284, b = 0.36669042706489563
Epoch 200: Loss = 0.012516680173575878, W = 1.927610993385315, b = 0.2613469958305359
Epoch 300: Loss = 0.006358101963996887, W = 1.9484070539474487, b = 0.18626676499843597
Epoch 400: Loss = 0.0032296902500092983, W = 1.9632288217544556, b = 0.13275568187236786
Epoch 500: Loss = 0.0016405789647251368, W = 1.973792552947998, b = 0.0946173295378685
Epoch 600: Loss = 0.0008333630976267159, W = 1.9813214540481567, b = 0.06743549555540085
Epoch 700: Loss = 0.0004233125655446201 6, W = 1.9866875410079956, b = 0.048062436282634735
Epoch 800: Loss = 0.00021502989693544805, W = 1.9905120134353638, b = 0.03425497189164162
Epoch 900: Loss = 0.00010923053196165711, W = 1.9932377338409424, b = 0.024414105340838432
Epoch 1000: Loss = 5.548369517782703e-05, W = 1.995180368423462, b = 0.017400488257408142
```

Hình. Kết quả huấn luyện

7.4.3.5. Bước 5: Dự đoán

Sử dụng mô hình đã huấn luyện để dự đoán trên dữ liệu mới.

```
# Dự đoán giá trị Y cho các giá trị X mới
X_new = np.array([6.0, 7.0], dtype=np.float32)
Y_new = predict(X_new)

print("Kết quả dự đoán:")
for i in range(len(X_new)):
 print(f"X = {X_new[i]}, Y = {Y_new.numpy()[i]}")
print (X, Y, '*')
print (X, Y_pred, 'r')
```



Hình. Đường dự báo

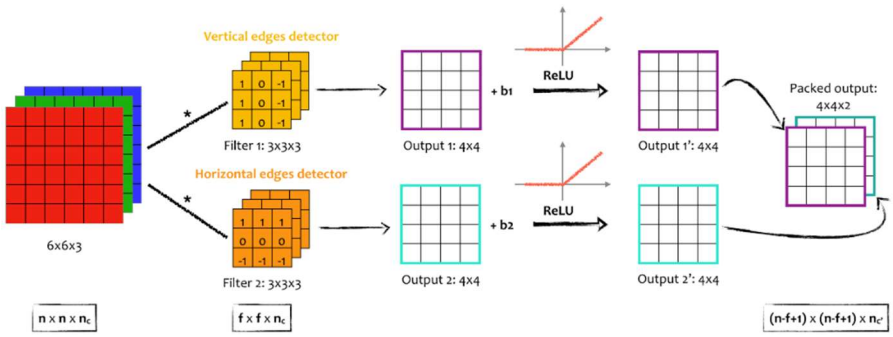
```
Kết quả dự đoán:
X = 6.0, Y = 11.988483428955078
X = 7.0, Y = 13.983663558959961
```

Hình. Kết quả dự báo

7.4.4. Mạng nơ ron tích chập

7.4.4.1. Khái niệm mạng tích chập

CNN là từ viết tắt của cụm Convolutional Neural Network hay là mạng nơ ron tích chập. Đây là mô hình vô cùng tiên tiến được áp dụng nhiều trong lĩnh vực học sâu. Mạng nơ ron này cho phép người dùng xây dựng những hệ thống phân loại và dự đoán với độ chính xác cực cao. Hiện nay, mạng CNN được ứng dụng nhiều hơn trong xử lý ảnh, cụ thể là nhận diện đối tượng trong hình.



Hình. Bài toán sử dụng mạng tích chập

7.4.4.2. Tích chập trong CNN

Đây là một "cửa sổ" sử dụng trượt trên ma trận nhằm lấy được những thông tin chính xác và cần thiết nhất mà không phải chọn đặc trưng (feature). Convolution hay nhân tích chập là cách mà những lớp Convolutional này nhân những phần tử trong ma trận. Sliding Window hay kernel là dạng ma trận có kích thước nhỏ, sử dụng trong nhân tích chập với ma trận hình ảnh.

7.4.4.3. Đặc trưng trong CNN

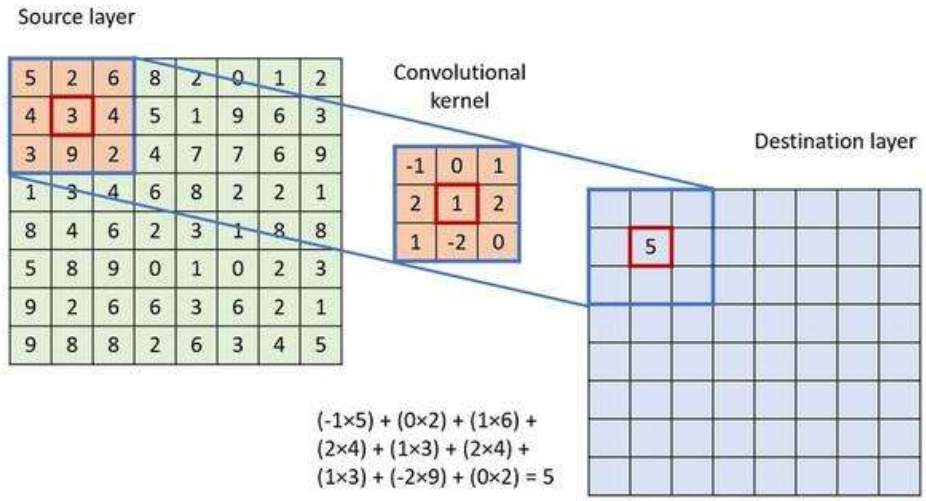
Feature là đặc trưng, mạng CNN sẽ so sánh dựa vào từng mảnh và các mảnh như vậy được gọi là feature. Thay vì phải tiến hành khớp các bức ảnh lại với nhau thì mạng nơ ron này sẽ xác định được sự tương đồng thông qua tìm kiếm thô những đặc trưng khớp với nhau bằng hai hình ảnh tốt hơn. Một feature là mộ hình ảnh dạng mini (những mảng 2 chiều nhỏ). Những feature này đều tương ứng với một khía cạnh nào đó của hình ảnh và chúng có thể khớp lại được với nhau.

Mạng CNN gồm những lớp cơ bản như trình bày sau đây.

7.4.4.4. Tầng tích chập

Lớp này là phần quan trọng nhất của toàn mạng CNN, nó có nhiệm vụ thực thi các tính toán. Các yếu tố quan trọng trong lớp Convolutional là: padding, stride, feature map và filter map.

1. Sử dụng **filter** để áp dụng vào các vùng của ma trận hình ảnh. Các filter map là các ma trận 3 chiều, bên trong đó là những tham số và chúng được gọi là parameters.
2. Stride tức là bạn dịch chuyển filter map theo từng pixel dựa vào các giá trị từ trái qua phải.
3. **Padding**: Thường thì giá trị viền xung quanh của ma trận hình ảnh sẽ được gán các giá trị 0 để có thể tiến hành nhân tích chập mà không làm giảm kích thước ma trận ảnh ban đầu.
4. **Feature map**: Biểu diễn kết quả sau mỗi lần feature map quét qua ma trận ảnh đầu vào. Sau mỗi lần quét thì lớp Convolutional sẽ tiến hành tính toán.



Hình. Tầng tích chập

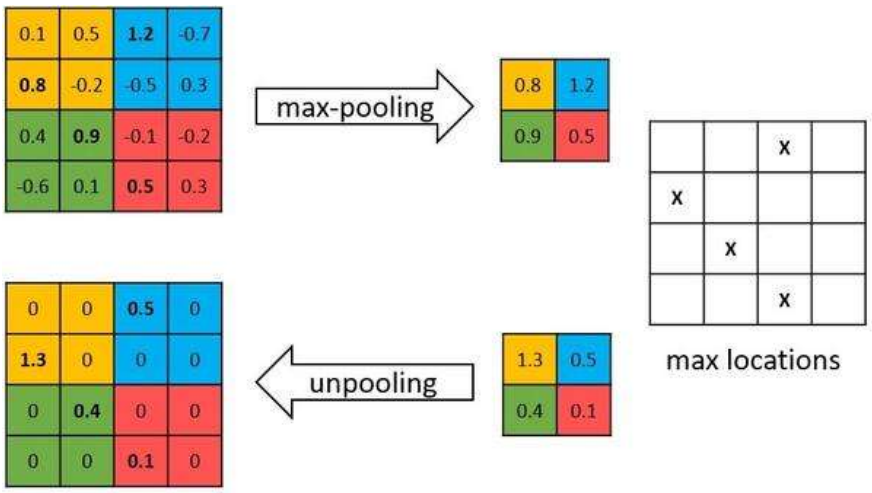
7.4.4.5. Tầng Relu

Lớp ReLU này là hàm kích hoạt trong mạng CNN, được gọi là activation function. Nó có tác dụng mô phỏng những nơ ron có tỷ lệ truyền xung qua axon. Các hàm activation khác như Leaky, Sigmoid, Leaky, Maxout,.. tuy nhiên hiện nay, hàm ReLU được sử dụng phổ biến và thông dụng nhất.

Hàm này được sử dụng cho những yêu cầu huấn luyện mạng nơ ron với những ưu điểm nổi bật điển hình là hỗ trợ tính toán nhanh hơn. Trong quá trình dùng hàm ReLU, bạn cần chú ý đến việc tùy chỉnh những learning rate và dead unit. Những lớp ReLU được dùng sau khi filter map được tính và áp dụng ReLU lên các giá trị của filter map.

7.4.4.6. Tầng Pooling

Khi ma trận ảnh đầu vào có kích thước quá lớn, các tầng Pooling sẽ được đặt vào giữa những lớp Convolutional để làm giảm những tham số. Hiện, hai loại lớp Pooling được sử dụng phổ biến là Max pooling và Average.



Hình. Tầng Pooling

7.4.4.7. Tầng kết nối toàn bộ

Đây là lớp có nhiệm vụ đưa ra kết quả sau khi hai lớp Convolutional và Pooling đaã nhận được ảnh truyền. Khi này, ta sẽ thu được một model đọc được thông tin của ảnh. Để có thể liên kế chúng cũng như cho nhiều đầu ra hơn ta sẽ sử dụng thuật ngữ kết nối toàn bộ.

Ngoài ra, nếu lớp này có dữ liệu hình ảnh thì lớp sẽ chuyển chúng thành các much chưa được phân chia chất lượng để tìm ra ảnh có chất lượng cao nhất.

7.4.4.8. Kiến trúc của mạng CNN

Mạng CNN là tập hợp những Convolutional layer xếp chồng lên nhau, đồng thời mạng sử dụng những hàm như ReLU và Tanh để kích hoạt các trọng số trong các node. Các lớp này sau khi qua các hàm activation sẽ có trọng số trong những node và có thể tạo ra những thông tin trừu tượng hơn đến với các lớp kế tiếp trong mạng.

Mạng này có tính kết hợp cà tính bất biến. Tức là, nếu cùng một đối tượng mà sử dụng chiếu theo các góc độ khác nhau thì sẽ có ảnh hưởng đến độ chính xác. Với dịch chuyển, co giãn hay quay ma trận ảnh thì lớp Pooling sẽ được dùng để hỗ trợ làm bất biến các tính chất này. Chính vì vậy mà mạng nơ ron này sẽ đưa ra những kết quả có độ chính xác tương ứng với từng mô hình.

Trong đó, lớp Pooling sẽ có khả năng tạo tính bất biến với phép dịch chuyển, co giãn và quay. Còn tính kết hợp cục bộ sẽ cho thấy những cấp độ

biểu diễn, dữ liệu từ thấp đến cao với mức trừu tượng thông qua Convolution từ filter. Mạng CNN có những lớp liên kết nhau dựa vào cơ chế Convolution.

Các lớp tiếp theo sẽ là kết quả từ những lớp trước đó, vì vậy mà bạn sẽ có những liên kết cục bộ phù hợp nhất. Trong quá trình huấn luyện mạng, mạng nơ ron này sẽ tự học hỏi những giá trị thông qua filter layer dựa theo cách thức người ta thực hiện.

Cấu trúc cơ bản của một mô hình mạng CNN thường bao gồm 3 phần chính bao gồm:

1. *Trường cục bộ*[1]: Lớp này sử dụng để tách lọc dữ liệu, thông tin hình ảnh để từ đó có thể lựa chọn các vùng có giá trị sử dụng hiệu quả cao nhất.
2. *Trọng số chia sẻ*[2]: Lớp này hỗ trợ làm giảm các tham số đến mức tối thiểu trong mạng CNN. Trong từng lớp convolution sẽ chứa các feature map riêng và từng feature thì sẽ có khả năng phát hiện một vài feature trong hình ảnh.
3. *Lớp tổng hợp*[3]: Đây là lớp cuối cùng và sử dụng để làm đơn giản các thông tin output. Tức là, sau khi tính toán xong và quét qua các layer trong mạng thì pooling layer sẽ được dùng để lược bỏ các thông tin không hữu ích. Từ đó cho ra kết quả theo kỳ vọng người dùng.

7.4.4.9. Cách lựa chọn tham số cho mạng CNN

Để chọn tham số phù hợp nhất cho mạng CNN, cần chú ý đến những yếu tố như: filter size, số convolution, pooling size và việc train – test.

1. *Lớp Convolution*: Số lượng lớp này càng nhiều thì sẽ giúp cải thiện được hoạt động của chương trình. Sử dụng những lớp với số lượng lớn thì khả năng hạn chế các tác động các tốt. Thông thường, chỉ sau khoảng 3 đến 4 lớp bạn sẽ đạt được kết quả như kỳ vọng
2. *Filter size*: Kích thước thường chọn là ma trận 3×3 hoặc ma trận 5×5.

[1] Local receptive field
[2] Shared weights and bias
[3] Pooling layer

3. *Pooling size*: Với những hình ảnh thông thường, bạn nên chọn ma trận pooling kích thước 2×2. Với những ảnh kích thước lớn thì nên chọn ma trận kích thước 3×3.
4. *Train – test*: Cần thực hiện train – test nhiều lần để có thể cho ra những parameter tốt nhất.

7.4.4.10. Thí dụ mạng CNN trong Jupyter

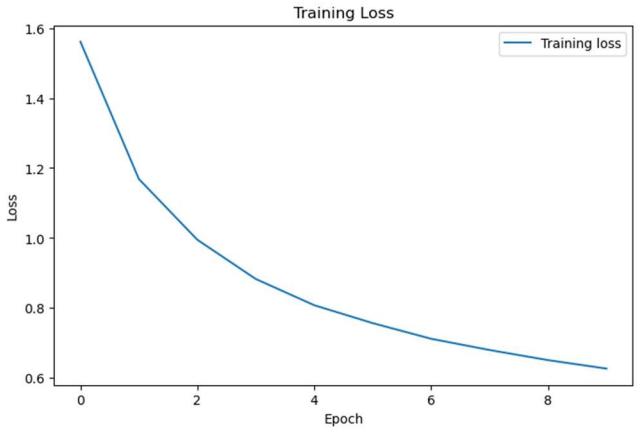
```
**Mạng noron tích chập nhận dạng thú vật, khuôn mặt**
import torch
import torch.nn as nn
import torch.optim as optim
import torchvision
import torchvision.transforms as transforms
import matplotlib.pyplot as plt

# Define transformations for the data
transform_train = transforms.Compose([
 transforms.RandomHorizontalFlip(),
 transforms.RandomCrop(32, padding=4),
 transforms.ToTensor(),
 transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))
])

transform_test = transforms.Compose([
 transforms.ToTensor(),
 transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))
])

# Download and load the CIFAR-10 dataset
trainset = torchvision.datasets.CIFAR10(root='./data',
train=True, download=True, transform=transform_train)
trainloader = torch.utils.data.DataLoader(trainset,
batch_size=64, shuffle=True)

testset = torchvision.datasets.CIFAR10(root='./data',
train=False, download=True, transform=transform_test)
testloader = torch.utils.data.DataLoader(testset,
batch_size=64, shuffle=False)

# Classes in CIFAR-10 dataset
classes = ('plane', 'car', 'bird', 'cat', 'deer', 'dog',
'frog', 'horse', 'ship', 'truck')
```

Tải https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz to./data\

```
100%|████████████████████████████████████████████████
████████████| 170498071/170498071 [06:55<00:00,
410426.45it/s]
```

```
 Extracting./data\cifar-10-python.tar.gz to./data
 Files already downloaded and verified

# Define the CNN model
class CNN(nn.Module):
 def __init__(self):
 super(CNN, self).__init__()
 self.conv1 = nn.Conv2d(3, 32, kernel_size=3, stride=1,
padding=1)
 self.relu1 = nn.ReLU()
 self.maxpool1 = nn.MaxPool2d(kernel_size=2, stride=2)
 self.conv2 = nn.Conv2d(32, 64, kernel_size=3, stride=1,
padding=1)
 self.relu2 = nn.ReLU()
 self.maxpool2 = nn.MaxPool2d(kernel_size=2, stride=2)
 self.conv3 = nn.Conv2d(64, 128, kernel_size=3, stride=1,
padding=1)
 self.relu3 = nn.ReLU()
 self.maxpool3 = nn.MaxPool2d(kernel_size=2, stride=2)
 self.flatten = nn.Flatten()
 self.fc1 = nn.Linear(128*4*4, 512)
 self.relu4 = nn.ReLU()
 self.fc2 = nn.Linear(512, 10)

 def forward(self, x):
 x = self.conv1(x)
 x = self.relu1(x)
 x = self.maxpool1(x)
 x = self.conv2(x)
 x = self.relu2(x)
 x = self.maxpool2(x)
 x = self.conv3(x)
 x = self.relu3(x)
 x = self.maxpool3(x)
 x = self.flatten(x)
 x = self.fc1(x)
 x = self.relu4(x)
 x = self.fc2(x)
 return x

# Initialize the model
model = CNN()

# Define loss function and optimizer
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)

# Training the model
num_epochs = 10
train_losses = []

for epoch in range(num_epochs):
```

```
 running_loss = 0.0
 for images, labels in trainloader:
 optimizer.zero_grad()
 outputs = model(images)
 loss = criterion(outputs, labels)
 loss.backward()
 optimizer.step()
 running_loss += loss.item()
 train_loss = running_loss / len(trainloader)
 train_losses.append(train_loss)
 print(f'Epoch [{epoch+1}/{num_epochs}], Loss:
{train_loss:.4f}')

 Epoch [1/10], Loss: 1.5610
 Epoch [2/10], Loss: 1.1686
 Epoch [3/10], Loss: 0.9950
 Epoch [4/10], Loss: 0.8829
 Epoch [5/10], Loss: 0.8080
 Epoch [6/10], Loss: 0.7567
 Epoch [7/10], Loss: 0.7120
 Epoch [8/10], Loss: 0.6799
 Epoch [9/10], Loss: 0.6510
 Epoch [10/10], Loss: 0.6267

# Evaluate the model
model.eval()
correct = 0
total = 0

with torch.no_grad():
 for images, labels in testloader:
 outputs = model(images)
 _, predicted = torch.max(outputs.data, 1)
 total += labels.size(0)
 correct += (predicted == labels).sum().item()

accuracy = correct / total
print(f'Accuracy on test set: {accuracy:.2%}')

# Plot training loss
plt.figure(figsize=(8, 5))
plt.plot(train_losses, label='Training loss')
plt.title('Training Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.show()
Accuracy on test set: 77.78%
```

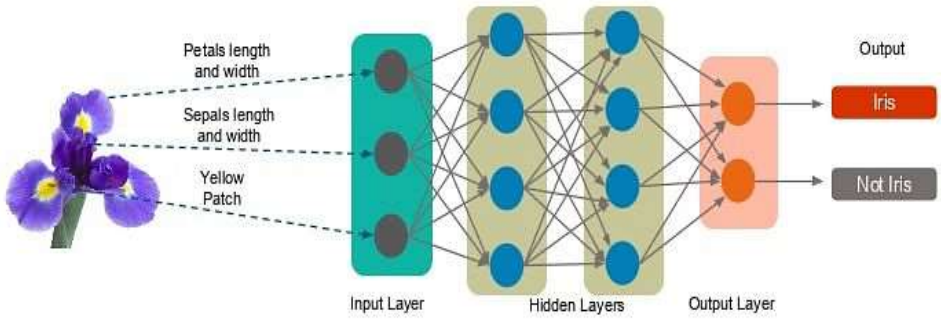


Hình. Kết quả của chương trình mạng CNN

7.4.5. Mạng nơ ron hồi qui

7.4.5.1. Giới thiệu

Mạng thần kinh có nhiều loại khác nhau, như Mạng nơ ron tích chập CNN, Mạng thần kinh nhân tạo ANN, RNN… RNN[1] là một loại mạng thần kinh, gọi là mạng nơ ron hồi qui.



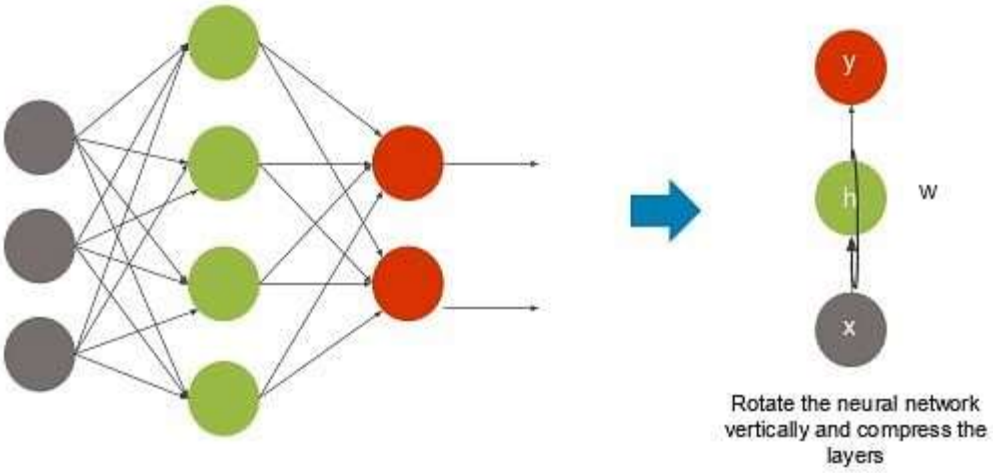
Hình. Các tầng trong mạng RNN

Các mạng như CNN và ANN là mạng chuyển tiếp nguồn cấp dữ liệu, trong đó thông tin chỉ đi từ trái sang phải hoặc từ trước ra sau. Với RNN, cũng có một số thông tin được truyền ngược lại. Và do đó nó được biết đến như một mạng lưới thần kinh tái phát.

Mỗi loại mạng lưới thần kinh có một ứng dụng cụ thể. Ví dụ: mạng thần kinh tích chập, hay CNN, rất lý tưởng để xử lý hình ảnh và phát hiện đối tượng cho video và hình ảnh. Mặt khác, RNN là một lựa chọn tuyệt vời cho Xử lý ngôn ngữ tự nhiên (NLP) hoặc nhận dạng giọng nói.

[1] Recurrent neural network RNN

Một RNN điển hình trông giống như hình ảnh hiển thị bên dưới:

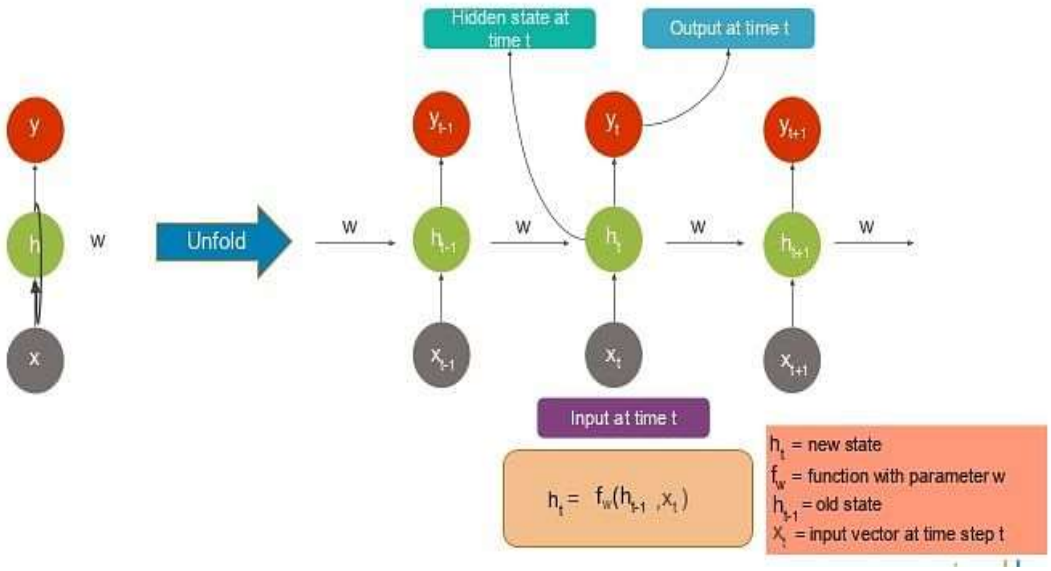


Hình. Mạng RNN điển hình

Các mạng như CNN và ANN là mạng chuyển tiếp nguồn cấp dữ liệu, trong đó thông tin chỉ đi từ trái sang phải hoặc từ trước ra sau. Với RNN, cũng có một số thông tin được truyền ngược lại. Và do đó nó được biết đến như một mạng lưới thần kinh tái phát.

Mỗi loại mạng lưới thần kinh có một ứng dụng cụ thể. Ví dụ: mạng thần kinh tích chập, hay CNN, rất lý tưởng để xử lý hình ảnh và phát hiện đối tượng cho video và hình ảnh. Mặt khác, RNN là một lựa chọn tuyệt vời cho Xử lý ngôn ngữ tự nhiên (NLP) hoặc nhận dạng giọng nói.

Một RNN điển hình trông giống như hình ảnh hiển thị bên dưới:



Hình. Mạng thần kinh hồi qui RNN

Lưu ý rằng đây không phải là ba tế bào thần kinh. Đây là một nơ-ron và nó được thể hiện theo cách chưa được mở ra.

Tại một thời điểm nhất định 't-1.'

Đầu vào của xt-1 được đưa đến nơ-ron và nó tạo ra đầu ra của yt-1

Khi chúng ta chuyển sang tức thời 't', nó sẽ chấp nhận đầu vào xt và đầu vào bổ sung từ khung thời gian 't-1' trước đó. Vì vậy, yt-1 cũng được cung cấp tại ht và điều đó dẫn đến yt

Tương tự với thời gian 't+1', đầu vào của xt+1 cộng với đầu vào từ khung thời gian trước đó xt và trạng thái ẩn tại ht+1 được cung cấp để có đầu ra là yt+1

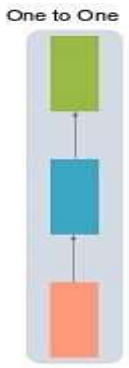
Hướng dẫn TensorFlow này cung cấp mô tả chi tiết về các loại RNN trong phần tiếp theo.

7.4.5.2. Các loại RNN

Có nhiều loại RNN khác nhau, tùy theo ứng dụng:

1. Một đối một
2. Một-nhiều
3. Nhiều-một
4. Nhiều-nhiều

Một-một : Nó được gọi là Mạng thần kinh Vanilla, dành cho các vấn đề về máy học thông thường. Có thể thấy RNN một-một bên dưới:

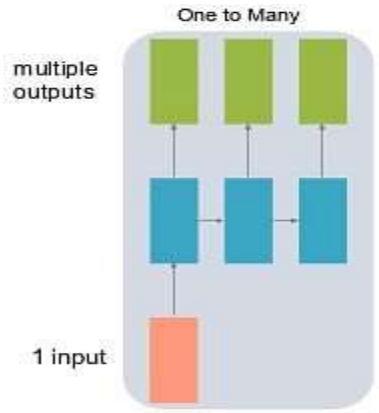


Hình. Một-một

Một ví dụ về mạng này có thể là giá cổ phiếu, trong đó chỉ cung cấp một 'giá cổ phiếu' đầu vào trải đều trong một thời gian. Sẽ nhận được kết quả

đầu ra, cũng chính là giá cổ phiếu, được dự đoán trong hai, ba hoặc mười ngày tới. Số lượng đầu ra và đầu vào là như nhau.

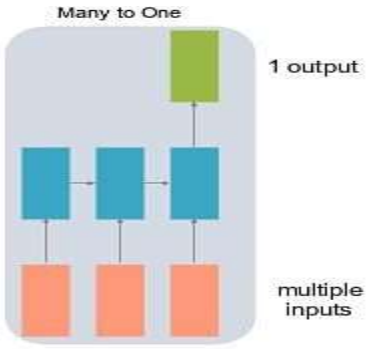
Một-nhiều : Ở đây, có một đầu vào và nhiều đầu ra. Mạng lưới như hình dưới đây:



Hình. Một- nhiều

Ví dụ: Giả sử muốn chú thích một hình ảnh. Đầu vào sẽ chỉ là một hình ảnh. Đối với đầu ra, đang tìm kiếm một cụm từ. Sau khi chia sẻ hình ảnh, mạng sẽ tạo ra một chuỗi các từ cho chú thích.

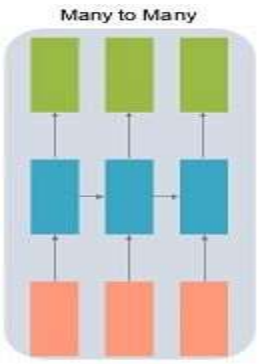
Nhiều-một: Loại mạng này được sử dụng để thực hiện phân tích tình cảm. Mạng lưới như hình dưới đây:



Hình. Nhiều-một

Ví dụ: đang cung cấp một số văn bản và muốn biết kết quả đầu ra là cảm xúc được thể hiện bằng văn bản. Nó có thể là tích cực hoặc tiêu cực. Đầu vào là nhiều từ khác nhau nhưng đầu ra chỉ có một từ.

Nhiều nhiều: Nó thường được sử dụng trong dịch máy. Mạng lưới như hình dưới đây:



Hình. Nhiều-nhiều

Ví dụ: Giả sử muốn dịch một số văn bản. Nhập một câu bằng một ngôn ngữ và sau đó muốn một câu khác bằng ngôn ngữ khác. Trong ví dụ này, đầu vào và đầu ra sẽ có nhiều từ. RNN rất hữu ích trong việc thực hiện phân tích chuỗi thời gian.

7.4.5.3. Triển khai ca sử dụng của RNN

Tuyên bố vấn đề - Chúng tôi đã thu thập dữ liệu liên quan đến sản xuất sữa trong vài tháng. Bằng cách sử dụng RNN, chúng tôi muốn dự đoán sản lượng sữa trên mỗi con bò tính bằng pound bằng cách sử dụng phân tích chuỗi thời gian.

Hướng dẫn TensorFlow này chỉ là phần giới thiệu về thế giới AI và khoa học dữ liệu vẫn đang phát triển. Hiểu các khái niệm khác về học sâu không phải là chuyện dễ dàng. Cần có hướng dẫn từng bước để hiểu những kiến thức cơ bản về học máy và học sâu.

7.4.6. Bộ nhớ dài ngắn hạn

7.4.6.1. Giới thiệu

Bộ nhớ LST là một thiết kế mạng thần kinh hồi qui RNN tiên tiến được phát triển để phản ánh chính xác hơn trình tự thời gian và các mối quan hệ ngắn gọn có liên quan. Các đặc điểm chính của nó bao gồm bố cục bên trong của ô LSTM, nhiều thay đổi được thực hiện đối với kiến trúc LSTM và một số triển khai LSTM theo yêu cầu.

7.4.6.2. Mạng LSTM

Mạng LSTM mở rộng mạng thần kinh tái phát (RNN) được thiết kế chủ yếu để xử lý các tình huống trong đó RNN không hoạt động. Khi chúng ta nói về RNN, nó là một thuật toán xử lý đầu vào hiện tại bằng cách tính đến đầu

ra của các sự kiện trước đó (phản hồi) và sau đó lưu trữ nó vào bộ nhớ của người dùng trong một khoảng thời gian ngắn (bộ nhớ ngắn hạn). Trong số nhiều ứng dụng, ứng dụng nổi tiếng nhất của nó là những ứng dụng trong lĩnh vực kiểm soát giọng nói và sáng tác nhạc phi Markovian. Tuy nhiên, có một số hạn chế đối với RNN.

Bộ nhớ ngắn hạn dài hạn (LSTM) đã được đưa vào hình ảnh vì đây là bộ nhớ đầu tiên không lưu được thông tin trong thời gian dài. Đôi khi cần có dữ liệu gốc được lưu trữ cách đây một thời gian đáng kể để xác định đầu ra của dữ liệu hiện tại. Tuy nhiên, RNN hoàn toàn không có khả năng quản lý những "sự phụ thuộc lâu dài" này.

Vấn đề thứ hai là không có sự kiểm soát nào tốt hơn về thành phần nào của bối cảnh cần được tiếp tục và phần nào của quá khứ phải bị lãng quên. Các vấn đề khác liên quan đến RNN là độ dốc bùng nổ hoặc biến mất (sẽ được giải thích sau) xảy ra trong quá trình huấn luyện RNN thông qua quá trình quay lui.

Do đó, vấn đề biến mất độ dốc gần như được loại bỏ hoàn toàn do mô hình đào tạo không bị ảnh hưởng. Độ trễ thời gian dài trong các vấn đề cụ thể được giải quyết bằng cách sử dụng LSTM, giải pháp này cũng xử lý các tác động của nhiễu, biểu diễn phân tán hoặc số vô tận.

Với LSTM, chúng không đáp ứng yêu cầu duy trì cùng số lượng trạng thái trước thời gian mà mô hình Markov ẩn náu (HMM) yêu cầu. LSTM cung cấp cho chúng ta một loạt các tham số như tốc độ học tập cũng như độ lệch đầu ra và đầu vào. Vì vậy, không cần thiết phải điều chỉnh nhỏ. Nỗ lực cập nhật từng trọng số được giảm xuống O(1) bằng cách sử dụng LSTM giống như các LSTM được sử dụng trong Lan truyền ngược xuyên thời gian (BPTT), đây là một lợi thế đáng kể.

7.4.6.3. Độ dốc bùng nổ và biến mất

Mục tiêu chính của việc huấn luyện mạng là giảm tổn thất ở đầu ra của mạng. Độ dốc hoặc mất mát với bộ trọng số được xác định để điều chỉnh trọng số và giảm thiểu tổn thất. Độ dốc trong một lớp phụ thuộc vào các khía cạnh của các lớp sau và nếu bất kỳ thành phần nào nhỏ sẽ dẫn đến độ dốc nhỏ hơn (hiệu ứng chia tỷ lệ).

Nhân hiệu ứng này với tốc độ học tập (0,1 đến 0,001) sẽ làm giảm sự thay đổi trọng lượng và tạo ra kết quả tương tự. Khi độ dốc đáng kể do các thành phần lớn, trọng lượng có thể thay đổi đáng kể, gây ra sự bùng nổ độ dốc. Để giải quyết các gradient bùng nổ, đơn vị mạng lưới thần kinh đã được xây dựng lại với hệ số tỷ lệ là một. Tế bào được tăng cường với các đơn vị gating, dẫn đến sự phát triển của LSTM.

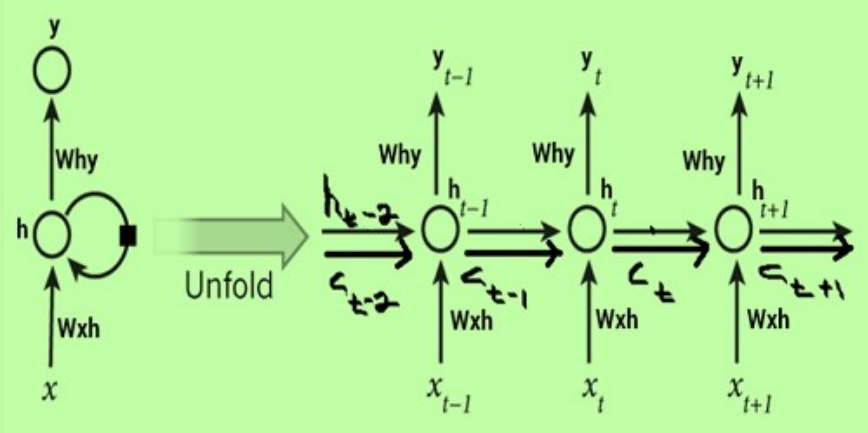
7.4.6.4. Kiến trúc của Mạng LSTM

Thiết kế của mạng LSTM (Bộ nhớ ngắn hạn) tương phản với RNN thông thường ở một số khía cạnh chính:

Cấu trúc lớp ẩn: Sự khác biệt chính giữa các cấu trúc bao gồm RNN cũng như LSTM có thể thấy ở chỗ lớp ẩn của LSTM là đơn vị hoặc ô được kiểm soát. Nó có bốn lớp hoạt động với nhau để tạo ra đầu ra của ô, cũng như trạng thái của ô. Cả hai đều được chuyển sang lớp tiếp theo.

Cơ chế cổng: Trái ngược với RNN, bao gồm lớp mạng lưới thần kinh duy nhất được tạo thành từ Tánh, LSTM bao gồm ba cổng sigmoid logistic và một lớp Tánh. Cổng đã được thêm vào để hạn chế thông tin đi qua các ô. Họ quyết định phần dữ liệu nào được yêu cầu trong ô tiếp theo và phần nào phải được loại bỏ. Đầu ra thường sẽ nằm trong phạm vi 0-1, trong đó "0" là tham chiếu đến "từ chối tất cả" trong khi "1" có nghĩa là "bao gồm tất cả".

Các lớp ẩn của LSTM: Mỗi ô LSTM được trang bị ba đầu vào và hai đầu ra, ht và Ct. Tại một thời điểm cụ thể, t, trong đó ht là trạng thái ẩn và Ct là trạng thái ô hoặc bộ nhớ. Chính xt là điểm thông tin hiện tại hoặc đầu vào. Lớp sigmoid đầu tiên chứa hai đầu vào: ht-1 và xt, trong đó ht-1 là trạng thái ẩn trong ô trước nó. Nó còn được biết đến với tên gọi và cổng quên vì đầu ra của nó là sự lựa chọn lượng dữ liệu từ ô cuối cùng cần được đưa vào. Đầu ra của nó sẽ là một số [0,1] nhân (theo chiều kim đồng hồ) với trạng thái của ô trước đó.



Hình. Mạng LSTM

7.4.6.5. Ứng dụng

Các mô hình LSTM phải được đào tạo bằng cách sử dụng tập dữ liệu huấn luyện trước khi sử dụng trong thế giới thực. Các ứng dụng thách thức nhất được liệt kê trong các phần sau:

1. Tạo văn bản: Tạo văn bản hoặc mô hình hóa ngôn ngữ liên quan đến việc tính toán các từ bất cứ khi nào một chuỗi từ được cung cấp làm đầu vào. Các mô hình ngôn ngữ có thể được sử dụng ở cấp độ ký tự hoặc cấp độ n-gram cũng như ở cấp độ câu hoặc cấp độ của một đoạn văn.
2. Xử lý hình ảnh: Các tổ chức LSTM có thể điều tra và mô tả hình ảnh bằng cách tạo ra các hình ảnh in. Ứng dụng này thường được sử dụng trong các dự án thị giác trên PC như phụ đề hình ảnh và nhận dạng đối tượng.
3. Nhận dạng giọng nói và chữ viết tay: Mạng LSTM có thể được sử dụng để nhận dạng và phiên âm ngôn ngữ nói hoặc văn bản viết tay. Ứng dụng này có ý nghĩa quan trọng trong hệ thống nhận dạng giọng nói và nhận dạng ký tự quang học (OCR).
4. Tạo nhạc: Mạng LSTM có thể tạo ra các chuỗi âm nhạc bằng cách học các mẫu từ dữ liệu âm nhạc hiện có. Ứng dụng này cho phép tạo ra các giai điệu, hòa âm và sáng tác mới.
5. Dịch ngôn ngữ: Mạng LSTM có thể được sử dụng trong các tác vụ dịch máy để chuyển đổi chuỗi văn bản từ ngôn ngữ này sang ngôn ngữ khác. Bằng cách học cách ánh xạ giữa các ngôn ngữ, mạng LSTM tạo điều kiện cho việc dịch ngôn ngữ tự động.

7.4.6.6. Nhược điểm của mạng LSTM

1. Thời gian đào tạo và cường độ tài nguyên: Mạng LSTM đào tạo có thể tăng cấp tính toán và tốn thời gian, đòi hỏi băng thông bộ nhớ cao, có thể gây lãng phí trong các thiết lập thiết bị cụ thể.
2. Cấu trúc tế bào phức tạp: Cấu trúc của các tế bào LSTM có thể trở nên phức tạp, khiến chúng khó hiểu và mổ xẻ hơn.
3. Thực thi quá mức và bỏ học: Mạng LSTM có thể có xu hướng phù hợp quá mức và việc thực hiện bỏ học, một quy trình chuẩn hóa, có thể được thử trong mạng LSTM.
4. Độ nhạy đối với việc khởi tạo ngẫu nhiên: LSTM có thể nhạy cảm với việc khởi tạo trọng lượng ngẫu nhiên, đòi hỏi phải khởi tạo thận trọng để tránh tình trạng mỏng manh trong quá trình đào tạo.
5. Bộ nhớ dài hạn hạn chế: LSTM có những hạn chế trong việc lưu trữ và truy cập bộ nhớ dài hạn, kích thích việc khám phá liên tục các mô hình có thể dễ dàng xử lý các điều kiện dài hạn hơn.

7.4.6.7. Ưu điểm của mạng LSTM

1. Mô hình hóa dữ liệu tuần tự hiệu quả: Mạng LSTM thành công trong việc mô hình hóa dữ liệu tuần tự và nắm bắt các điều kiện tầm xa, khiến chúng phù hợp với các dự án, chẳng hạn như mô hình hóa ngôn ngữ và nhận dạng diễn ngôn.
2. Tính linh hoạt của ứng dụng: Mạng LSTM có các ứng dụng trong các lĩnh vực khác nhau, bao gồm xử lý ngôn ngữ thông thường, thị giác PC, nhận dạng diễn ngôn, thời đại âm nhạc và giải thích ngôn ngữ.
3. Xử lý sự phụ thuộc dài hạn: Ngược lại với RNN thông thường, mạng LSTM nhằm mục đích xử lý các điều kiện dài hạn hiệu quả hơn, cho phép chúng lưu giữ và sử dụng dữ liệu kế thừa quá mức.
4. Cải thiện dòng chuyển màu: LSTM kiểm soát vấn đề biến mất độ dốc, xem xét việc đào tạo mạng lưới não sâu sắc một cách ổn định và hiệu quả hơn.
5. Những tiến bộ liên tục: Tiến trình công việc đổi mới trong mạng LSTM tiếp tục nỗ lực về cách trình bày, tính hiệu quả và khả năng diễn giải của chúng, đưa ra những tiến bộ như hệ thống cân nhắc và thiết kế lai.

Nhìn chung, mạng LSTM đã trở thành một thiết bị quan trọng trong AI nhờ khả năng hiển thị thông tin liên tục và nắm bắt các điều kiện đường dài. Họ đã tìm thấy các ứng dụng trong các không gian khác nhau, bao gồm xử lý ngôn ngữ thông thường, thị giác PC, nhận dạng diễn ngôn, thời đại âm nhạc và giải thích ngôn ngữ. Mặc dù mạng LSTM có những nhược điểm, nhưng việc phát triển công việc đổi mới đồng nghĩa với việc giải quyết những hạn chế này và tiếp tục phát huy khả năng của các mô hình dựa trên LSTM.

7.4.6.8. Thí dụ phân tích dữ liệu chuỗi thời gian với mạng LSTM

Bộ dữ liệu có các thuộc tính sau:

- 1.date: Ngày ở định dạng dd/mm/yyyy
- 2.time: thời gian ở định dạng hh:mm:ss
- 3.global_active_power: công suất hoạt động trung bình theo phút toàn cầu của hộ gia đình (tính bằng kilowatt)
- 4.global_reactive_power: công suất phản kháng trung bình theo phút toàn cầu của hộ gia đình (tính bằng kilowatt)
- 5.điện áp: điện áp trung bình phút (tính bằng volt)
- 6.global_intensity: cường độ dòng điện trung bình theo phút toàn cầu của hộ gia đình (tính bằng ampe)
- 7.sub_metering_1: đo lường phụ năng lượng số 1 (tính theo watt-giờ năng lượng hoạt động). Nó tương ứng với nhà bếp, chủ yếu chứa máy rửa chén, lò nướng và lò vi sóng (bếp nóng không chạy bằng điện mà chạy bằng gas).
- 8.sub_metering_2: đo lường phụ năng lượng số 2 (tính theo watt-giờ năng lượng hoạt động). Nó tương ứng với phòng giặt, có máy giặt, máy sấy quần áo, tủ lạnh và đèn.
- 9.sub_metering_3: đo lường phụ năng lượng số 3 (tính theo watt-giờ năng lượng hoạt động). Nó tương ứng với một máy nước nóng điện và một máy điều hòa không khí.

Chương trình trong Jupyter như sau.

```
In [1]:
# Let`s import all packages that we may need:

import sys
import numpy as np # linear algebra
```

```
from scipy.stats import randint
import pandas as pd # data processing, CSV file I/O (e.g.
pd.read_csv), data manipulation as in SQL
import matplotlib.pyplot as plt # this is used for the
plot the graph
import seaborn as sns # used for plot interactive graph.
from sklearn.model_selection import train_test_split # to
split the data into two parts
from sklearn.cross_validation import KFold # use for
cross validation
from sklearn.preprocessing import StandardScaler # for
normalization
from sklearn.preprocessing import MinMaxScaler
from sklearn.pipeline import Pipeline # pipeline making
from sklearn.model_selection import cross_val_score
from sklearn.feature_selection import SelectFromModel
from sklearn import metrics # for the check the error and
accuracy of the model
from sklearn.metrics import mean_squared_error,r2_score

## for Deep-learing:
import keras
from keras.layers import Dense
from keras.models import Sequential
from keras.utils import to_categorical
from keras.optimizers import SGD
from keras.callbacks import EarlyStopping
from keras.utils import np_utils
import itertools
from keras.layers import LSTM
from keras.layers.convolutional import Conv1D
from keras.layers.convolutional import MaxPooling1D
from keras.layers import Dropout
/opt/conda/lib/python3.6/site-
packages/sklearn/cross_validation.py:41:
DeprecationWarning: This module was deprecated in version
0.18 in favor of the model_selection module into which
all the refactored classes and functions are moved. Also
note that the interface of the new CV iterators are
different from that of this module. This module will be
removed in 0.20.
 "This module will be removed in 0.20.",
DeprecationWarning)
Using TensorFlow backend.
In [2]:
## Data can be downloaded from:
http://archive.ics.uci.edu/ml/machine-learning-
databases/00235/
## Just open the zip file and grab the file
'household_power_consumption.txt' put it in the directory
## that you would like to run the code.
```

```
df =
pd.read_csv('../input/household_power_consumption.txt',
sep=';',
 parse_dates={'dt' : ['Date', 'Time']},
infer_datetime_format=True,
 low_memory=False, na_values=['nan','?'], index_col='dt')
```

1. Lưu ý rằng dữ liệu bao gồm 'nan' và '?' như một chuỗi. Tôi đã chuyển đổi cả hai thành nanpy trong giai đoạn nhập (ở trên) và xử lý cả hai như nhau.
2. Tôi đã hợp nhất hai cột 'Ngày' và 'Thời gian' thành 'dt'.
3. Tôi cũng đã chuyển đổi ở trên, dữ liệu thành loại chuỗi thời gian, bằng cách lấy chỉ mục làm thời gian.In [3]:

```
df.head()
Out[3]:
```

	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	Sub_metering_2	Sub_metering_3
dt							
2006-12-16 17:24:00	4.216	0.418	234.84	18.4	0.0	1.0	17.0
2006-12-16 17:25:00	5.360	0.436	233.63	23.0	0.0	1.0	16.0
2006-12-16 17:26:00	5.374	0.498	233.29	23.0	0.0	2.0	17.0
2006-12-16 17:27:00	5.388	0.502	233.74	23.0	0.0	1.0	17.0
2006-12-16 17:28:00	3.666	0.528	235.68	15.8	0.0	1.0	17.0

```
In [4]:
df.info()
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 2075259 entries, 2006-12-16 17:24:00 to
2010-11-26 21:02:00
Data columns (total 7 columns):
Global_active_power float64
Global_reactive_power float64
Voltage float64
Global_intensity float64
Sub_metering_1 float64
Sub_metering_2 float64
Sub_metering_3 float64
dtypes: float64(7)
memory usage: 126.7 MB
In [5]:
df.dtypes
Out[5]:
Global_active_power float64
Global_reactive_power float64
Voltage float64
Global_intensity float64
```

```
Sub_metering_1 float64
Sub_metering_2 float64
Sub_metering_3 float64
dtype: object
In [6]:
df.shape
Out[6]:
(2075259, 7)
In [7]:
df.describe()
Out[7]:
```

	Global_active_power	Global_reactive_power	Voltage	Global_intensity	Sub_metering_1	Sub_metering_2	Sub_metering_3
count	2.049280e+06	2.049280e+06	2.049280e+06	2.049280e+06	2.049280e+06	2.049280e+06	2.049280e+06
mean	1.091615e+00	1.237145e-01	2.408399e+02	4.627759e+00	1.121923e+00	1.298520e+00	6.458447e+00
std	1.057294e+00	1.127220e-01	3.239987e+00	4.444396e+00	6.153031e+00	5.822026e+00	8.437154e+00
min	7.600000e-02	0.000000e+00	2.232000e+02	2.000000e-01	0.000000e+00	0.000000e+00	0.000000e+00
25%	3.080000e-01	4.800000e-02	2.389900e+02	1.400000e+00	0.000000e+00	0.000000e+00	0.000000e+00
50%	6.020000e-01	1.000000e-01	2.410100e+02	2.600000e+00	0.000000e+00	0.000000e+00	1.000000e+00
75%	1.528000e+00	1.940000e-01	2.428900e+02	6.400000e+00	0.000000e+00	1.000000e+00	1.700000e+01
max	1.112200e+01	1.390000e+00	2.541500e+02	4.840000e+01	8.800000e+01	8.000000e+01	3.100000e+01

```
In [8]:
df.columns
Out[8]:
Index(['Global_active_power', 'Global_reactive_power',
'Voltage',
 'Global_intensity', 'Sub_metering_1', 'Sub_metering_2',
 'Sub_metering_3'],
 dtype='object')
Dealing with missing values 'nan' with a test statistic
In [9]:
## finding all columns that have nan:

droping_list_all=[]
for j in range(0,7):
 if not df.iloc[:, j].notnull().all():
 droping_list_all.append(j)
 #print(df.iloc[:,j].unique())
droping_list_all
Out[9]:
[0, 1, 2, 3, 4, 5, 6]
In [10]:
```

```
# filling nan with mean in any columns

for j in range(0,7):
 df.iloc[:,j]=df.iloc[:,j].fillna(df.iloc[:,j].mean())
In [11]:
# another sanity check to make sure that there are not
more any nan
df.isnull().sum()
Out[11]:
Global_active_power 0
Global_reactive_power 0
Voltage 0
Global_intensity 0
Sub_metering_1 0
Sub_metering_2 0
Sub_metering_3 0
dtype: int64
```

4. Trực quan hóa dữ liệu

Bên dưới tôi lấy mẫu lại theo ngày và hiển thị tổng và giá trị trung bình của Global_active_power. Người ta thấy rằng giá trị trung bình và tổng của tập dữ liệu được lấy mẫu lại có cấu trúc tương tự nhau.

```
In [12]:
df.Global_active_power.resample('D').sum().plot(title='Gl
obal_active_power resampled over day for sum')
#df.Global_active_power.resample('D').mean().plot(title='
Global_active_power resampled over day', color='red')
plt.tight_layout()
plt.show()

df.Global_active_power.resample('D').mean().plot(title='G
lobal_active_power resampled over day for mean',
color='red')
plt.tight_layout()
plt.show()
In [13]:
### Below I show mean and std of 'Global_intensity'
resampled over day
r = df.Global_intensity.resample('D').agg(['mean',
'std'])
r.plot(subplots = True, title='Global_intensity resampled
over day')
plt.show()

In [14]:
### Below I show mean and std of 'Global_reactive_power'
resampled over day
r2 = df.Global_reactive_power.resample('D').agg(['mean',
'std'])
```

```
r2.plot(subplots = True, title='Global_reactive_power
resampled over day', color='red')
plt.show()

In [15]:
### Sum of 'Global_active_power' resampled over month
# Sum of 'Global_active_power' resampled over month
df['Global_active_power'].resample('M').mean().plot(kind=
'bar')
plt.xticks(rotation=60)
plt.ylabel('Global_active_power')
plt.title('Global_active_power per month (averaged over
month)')
plt.show()

In [16]:
## Mean of 'Global_active_power' resampled over quarter
df['Global_active_power'].resample('Q').mean().plot(kind=
'bar')
plt.xticks(rotation=60)
plt.ylabel('Global_active_power')
plt.title('Global_active_power per quarter (averaged over
quarter)')
plt.show()
```

Điều rất quan trọng cần lưu ý từ hai sơ đồ trên là việc lấy mẫu lại trong khoảng thời gian lớn hơn sẽ làm giảm tính tuần hoàn của hệ thống như chúng ta mong đợi. Điều này rất quan trọng đối với kỹ thuật tính năng học máy.

```
In [17]:
## mean of 'Voltage' resampled over month
df['Voltage'].resample('M').mean().plot(kind='bar',
color='red')
plt.xticks(rotation=60)
plt.ylabel('Voltage')
plt.title('Voltage per quarter (summed over quarter)')
plt.show()

In [18]:
df['Sub_metering_1'].resample('M').mean().plot(kind='bar'
, color='brown')
plt.xticks(rotation=60)
plt.ylabel('Sub_metering_1')
plt.title('Sub_metering_1 per quarter (summed over
quarter)')
plt.show()
* Từ đồ thị trên cho thấy giá trị trung bình của 'Điện
áp' trong tháng gần như không đổi so với các đặc điểm
khác. Điều này lại quan trọng trong việc lựa chọn tính
năng.
```

```
In [19]:
# Below I compare the mean of different featuresresampled
over day.
# specify columns to plot
cols = [0, 1, 2, 3, 5, 6]
i = 1
groups=cols
values = df.resample('D').mean().values
# plot each column
plt.figure(figsize=(15, 10))
for group in groups:
      plt.subplot(len(cols), 1, i)
      plt.plot(values[:, group])
      plt.title(df.columns[group], y=0.75, loc='right')
      i += 1
plt.show()

In [20]:
## resampling over week and computing mean
df.Global_reactive_power.resample('W').mean().plot(color=
'y', legend=True)
df.Global_active_power.resample('W').mean().plot(color='r
', legend=True)
df.Sub_metering_1.resample('W').mean().plot(color='b',
legend=True)
df.Global_intensity.resample('W').mean().plot(color='g',
legend=True)
plt.show()

In [21]:
# Below I show hist plot of the mean of different feature
resampled over month
df.Global_active_power.resample('M').mean().plot(kind='hi
st', color='r', legend=True )
df.Global_reactive_power.resample('M').mean().plot(kind='
hist',color='b', legend=True)
#df.Voltage.resample('M').sum().plot(kind='hist',color='g
', legend=True)
df.Global_intensity.resample('M').mean().plot(kind='hist'
, color='g', legend=True)
df.Sub_metering_1.resample('M').mean().plot(kind='hist',
color='y', legend=True)
plt.show()

In [22]:
## The correlations between 'Global_intensity',
'Global_active_power'
data_returns = df.pct_change()
sns.jointplot(x='Global_intensity',
y='Global_active_power', data=data_returns)

plt.show()
```

- Từ hai biểu đồ trên, có thể thấy rằng 'Global_intensity' và 'Global_active_power' có mối tương quan với nhau. Nhưng 'Điện áp', 'Global_active_power' ít tương quan hơn. Đây là quan sát quan trọng cho mục đích học máy.

```
In [23]:
## The correlations between 'Voltage' and
'Global_active_power'
sns.jointplot(x='Voltage', y='Global_active_power',
data=data_returns)
plt.show()
```

```
5. Tương quan giữa các đặc tính
In [24]:
# Correlations among columns
plt.matshow(df.corr(method='spearman'),vmax=1,vmin=-
1,cmap='PRGn')
plt.title('without resampling', size=15)
plt.colorbar()
plt.show()
```

```
In [25]:
# Correlations of mean of features resampled over months

plt.matshow(df.resample('M').mean().corr(method='spearman
'),vmax=1,vmin=-1,cmap='PRGn')
plt.title('resampled over month', size=15)
plt.colorbar()
plt.margins(0.02)
plt.matshow(df.resample('A').mean().corr(method='spearman
'),vmax=1,vmin=-1,cmap='PRGn')
plt.title('resampled over year', size=15)
plt.colorbar()
plt.show()
```

- Từ trên có thể thấy rằng với kỹ thuật lấy mẫu lại, người ta có thể thay đổi mối tương quan giữa các đặc điểm. Điều này rất quan trọng đối với kỹ thuật tính năng.

6. Nghiên cứu máy: Chuẩn bị dữ liệu LSTM và kỹ thuật tính năng

- Tôi sẽ áp dụng mạng số hồi quy (LSTM) phù hợp nhất cho bài toán chuỗi thời gian và tuần tự. Cách tiếp cận này là tốt nhất nếu chúng ta có dữ liệu lớn.
- Tôi sẽ trình bày vấn đề học có giám sát dưới dạng dự đoán Global_active_power tại thời điểm hiện tại (t) dựa trên phép đo

Global_active_power và các tính năng khác ở bước thời gian trước đó.

```
In [26]:
def series_to_supervised(data, n_in=1, n_out=1,
dropnan=True):
      n_vars = 1 if type(data) is list else data.shape[1]
      dff = pd.DataFrame(data)
      cols, names = list(), list()
      # input sequence (t-n,... t-1)
      for i in range(n_in, 0, -1):
            cols.append(dff.shift(i))
            names += [('var%d(t-%d)' % (j+1, i)) for j in
range(n_vars)]
      # forecast sequence (t, t+1,... t+n)
      for i in range(0, n_out):
            cols.append(dff.shift(-i))
            if i == 0:
                  names += [('var%d(t)' % (j+1)) for j in
range(n_vars)]
            else:
                  names += [('var%d(t+%d)' % (j+1, i)) for
j in range(n_vars)]
      # put it all together
      agg = pd.concat(cols, axis=1)
      agg.columns = names
      # drop rows with NaN values
      if dropnan:
            agg.dropna(inplace=True)
      return agg
```

Nhằm giảm thời gian tính toán, đồng thời có kết quả nhanh để kiểm định mô hình. Người ta có thể sắp xếp lại dữ liệu theo giờ (dữ liệu gốc được tính bằng phút). Điều này sẽ giảm kích thước dữ liệu từ 2075259 xuống 34589 nhưng vẫn giữ cấu trúc tổng thể của dữ liệu như ở trên.

```
In [27]:
## resampling of data over hour
df_resample = df.resample('h').mean()
df_resample.shape
Out[27]:
(34589, 7)
In [28]:
## * Note: I scale all features in range of [0,1].

## If you would like to train based on the resampled data
(over hour), then used below
values = df_resample.values
```

```
## full data without resampling
#values = df.values

# integer encode direction
# ensure all data is float
#values = values.astype('float32')
# normalize features
scaler = MinMaxScaler(feature_range=(0, 1))
scaled = scaler.fit_transform(values)
# frame as supervised learning
reframed = series_to_supervised(scaled, 1, 1)

# drop columns we don't want to predict
reframed.drop(reframed.columns[[8,9,10,11,12,13]],
axis=1, inplace=True)
print(reframed.head())
 var1(t-1) var2(t-1) var3(t-1) var4(t-1) var5(t-1)
var6(t-1) \
1 0.636816 0.295738 0.337945 0.631157 0.0 0.011366
2 0.545045 0.103358 0.335501 0.541487 0.0 0.144652
3 0.509006 0.110073 0.283802 0.502152 0.0 0.030869
4 0.488550 0.096987 0.315987 0.481110 0.0 0.000000
5 0.455597 0.099010 0.434417 0.449904 0.0 0.008973

 var7(t-1) var1(t)
1 0.782418 0.545045
2 0.782676 0.509006
3 0.774169 0.488550
4 0.778809 0.455597
5 0.798917 0.322555
```

Ở trên tôi đã hiển thị 7 biến đầu vào (chuỗi đầu vào) và 1 biến đầu ra cho 'Global_active_power' tại thời điểm hiện tại tính bằng giờ (tùy thuộc vào việc lấy mẫu lại).

7. Chia phần còn lại của dữ liệu thành tập huấn luyện và xác thực

Đầu tiên, tôi chia tập dữ liệu đã chuẩn bị thành tập huấn luyện và tập kiểm tra. Để tăng tốc quá trình đào tạo mô hình (để trình diễn), chúng tôi sẽ chỉ đào tạo mô hình trong năm dữ liệu đầu tiên, sau đó đánh giá mô hình trong 3 năm dữ liệu tiếp theo.

```
In [29]:
# split into train and test sets
values = reframed.values

n_train_time = 365*24
train = values[:n_train_time, :]
test = values[n_train_time:, :]
##test = values[n_train_time:n_test_time, :]
# split into input and outputs
train_X, train_y = train[:, :-1], train[:, -1]
```

```
test_X, test_y = test[:, :-1], test[:, -1]
# reshape input to be 3D [samples, timesteps, features]
train_X = train_X.reshape((train_X.shape[0], 1,
train_X.shape[1]))
test_X = test_X.reshape((test_X.shape[0], 1,
test_X.shape[1]))
print(train_X.shape, train_y.shape, test_X.shape,
test_y.shape)
# We reshaped the input into the 3D format as expected by
LSTMs, namely [samples, timesteps, features].
(8760, 1, 7) (8760,) (25828, 1, 7) (25828,)
```

8. Kiến trúc mô hình

1. LSTM với 100 nơ-ron ở lớp hiển thị đầu tiên
2. bỏ học 20%
3. 1 nơ-ron ở lớp đầu ra để dự đoán Global_active_power.
4. Hình dạng đầu vào sẽ là 1 bước thời gian với 7 tính năng.
5. Tôi sử dụng hàm mất Sai số tuyệt đối trung bình (MAE) và phiên bản Adam hiệu quả của việc giảm độ dốc ngẫu nhiên.
6. Mô hình sẽ phù hợp với 20 giai đoạn đào tạo với quy mô lô là 70.In [30]:

```
model = Sequential()
model.add(LSTM(100, input_shape=(train_X.shape[1],
train_X.shape[2])))
model.add(Dropout(0.2))
# model.add(LSTM(70))
# model.add(Dropout(0.3))
model.add(Dense(1))
model.compile(loss='mean_squared_error',
optimizer='adam')

# fit network
history = model.fit(train_X, train_y, epochs=20,
batch_size=70, validation_data=(test_X, test_y),
verbose=2, shuffle=False)

# summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper right')
plt.show()

# make a prediction
yhat = model.predict(test_X)
test_X = test_X.reshape((test_X.shape[0], 7))
# invert scaling for forecast
```

```
inv_yhat = np.concatenate((yhat, test_X[:, -6:]), axis=1)
inv_yhat = scaler.inverse_transform(inv_yhat)
inv_yhat = inv_yhat[:,0]
# invert scaling for actual
test_y = test_y.reshape((len(test_y), 1))
inv_y = np.concatenate((test_y, test_X[:, -6:]), axis=1)
inv_y = scaler.inverse_transform(inv_y)
inv_y = inv_y[:,0]
# calculate RMSE
rmse = np.sqrt(mean_squared_error(inv_y, inv_yhat))
print('Test RMSE: %.3f' % rmse)
Train on 8760 samples, validate on 25828 samples
Epoch 1/20
 - 1s - loss: 0.0189 - val_loss: 0.0122
Epoch 2/20
 - 1s - loss: 0.0126 - val_loss: 0.0106
Epoch 3/20
 - 1s - loss: 0.0114 - val_loss: 0.0096
Epoch 4/20
 - 1s - loss: 0.0108 - val_loss: 0.0094
Epoch 5/20
 - 1s - loss: 0.0106 - val_loss: 0.0092
Epoch 6/20
 - 1s - loss: 0.0106 - val_loss: 0.0093
Epoch 7/20
 - 1s - loss: 0.0106 - val_loss: 0.0093
Epoch 8/20
 - 1s - loss: 0.0105 - val_loss: 0.0093
Epoch 9/20
 - 1s - loss: 0.0106 - val_loss: 0.0092
Epoch 10/20
 - 1s - loss: 0.0105 - val_loss: 0.0093
Epoch 11/20
 - 1s - loss: 0.0104 - val_loss: 0.0093
Epoch 12/20
 - 1s - loss: 0.0104 - val_loss: 0.0092
Epoch 13/20
 - 1s - loss: 0.0104 - val_loss: 0.0092
Epoch 14/20
 - 1s - loss: 0.0104 - val_loss: 0.0092
Epoch 15/20
 - 1s - loss: 0.0104 - val_loss: 0.0093
Epoch 16/20
 - 1s - loss: 0.0103 - val_loss: 0.0092
Epoch 17/20
 - 1s - loss: 0.0104 - val_loss: 0.0094
Epoch 18/20
 - 1s - loss: 0.0104 - val_loss: 0.0093
Epoch 19/20
 - 1s - loss: 0.0103 - val_loss: 0.0092
Epoch 20/20
 - 1s - loss: 0.0104 - val_loss: 0.0093
```

```
Test RMSE: 0.620
```

Lưu ý rằng để cải thiện mô hình, người ta phải điều chỉnh epoch và batch_size.

```
In [31]:
## time steps, every step is one hour (you can easily
convert the time step to the actual time index)
## for a demonstration purpose, I only compare the
predictions in 200 hours.
aa=[x for x in range(200)]
plt.plot(aa, inv_y[:200], marker='.', label="actual")
plt.plot(aa, inv_yhat[:200], 'r', label="prediction")
plt.ylabel('Global_active_power', size=15)
plt.xlabel('Time step', size=15)
plt.legend(fontsize=15)
plt.show()
```

9. Nhận xét cuối cùng

- Ở đây tôi đã sử dụng mạng nơ-ron LSTM hiện là công nghệ tiên tiến nhất cho các bài toán tuần tự.
- Để giảm thời gian tính toán và nhanh chóng thu được một số kết quả, tôi đã lấy dữ liệu của năm đầu tiên (được lấy mẫu lại theo giờ) để huấn luyện mô hình và phần dữ liệu còn lại để kiểm tra mô hình.
- Tôi đã tập hợp một mạng lưới thần kinh LSTM rất đơn giản để chứng minh rằng người ta có thể đạt được những dự đoán hợp lý. Tuy nhiên, số lượng hàng quá cao và kết quả là việc tính toán rất tốn thời gian (ngay cả đối với mô hình đơn giản ở trên cũng mất vài phút để chạy trên Intel Core i7 2,8 GHz). Tốt nhất là viết phần mã cuối cùng bằng Spark (MLlib) chạy trên GPU.
- Hơn nữa, kiến trúc mạng nơ-ron mà tôi thiết kế chỉ là một mô hình đồ chơi. Nó có thể được cải thiện dễ dàng bằng cách thêm CNN và các lớp bỏ học. CNN rất hữu ích ở đây vì có mối tương quan trong dữ liệu (lớp CNN là một cách tốt để thăm dò cấu trúc dữ liệu cục bộ).

7.4.7. Seq2seq

Trong học sâu, mô hình tuần tự (seq2seq) là một loại kiến trúc mạng thần kinh được thiết kế để xử lý các tác vụ trong đó đầu vào và đầu ra đều là chuỗi. Chúng đặc biệt hữu ích cho các bài toán trong đó độ dài của chuỗi đầu vào và chuỗi đầu ra có thể khác nhau.

7.4.7.1. Các thành phần chính của Mô hình Seq2Seq:

Mã hoá:

1. Bộ mã hóa xử lý chuỗi đầu vào và nén nó thành vectơ ngữ cảnh có kích thước cố định (hoặc một tập hợp vectơ nếu sử dụng kiến trúc nâng cao hơn).
2. Nó thường bao gồm các mạng thần kinh tái phát (RNN), chẳng hạn như mạng Bộ nhớ ngắn hạn dài (LSTM) hoặc Đơn vị tái phát có kiểm soát (GRU), có khả năng nắm bắt các phụ thuộc tạm thời trong dữ liệu đầu vào.

Bộ giải mã:

1. Bộ giải mã lấy vectơ ngữ cảnh (hoặc các vectơ) từ bộ mã hóa và tạo ra chuỗi đầu ra theo từng bước.
2. Giống như bộ mã hóa, bộ giải mã cũng có thể sử dụng RNN và nó thường có cơ chế chú ý để tập trung vào các phần khác nhau của chuỗi đầu vào cho mỗi bước của chuỗi đầu ra.

Cơ chế chú ý (tùy chọn nhưng thường được sử dụng):

1. Cơ chế chú ý cho phép bộ giải mã tập trung có chọn lọc vào các phần khác nhau của chuỗi đầu vào ở mỗi bước của chuỗi đầu ra. Điều này hữu ích cho các tác vụ như dịch thuật, trong đó các phần khác nhau của chuỗi đầu vào có liên quan tại các thời điểm khác nhau.
2. Sự chú ý có thể giúp mô hình xử lý các chuỗi dài hơn và cải thiện hiệu suất bằng cách cho phép mô hình sử dụng thông tin từ tất cả các phần của chuỗi đầu vào, thay vì nén nó vào một vectơ ngữ cảnh duy nhất.

7.4.7.2. Các ứng dụng của Mô hình Seq2Seq

- Dịch máy: Chuyển đổi văn bản từ ngôn ngữ này sang ngôn ngữ khác. Ví dụ: dịch một câu tiếng Anh sang tiếng Pháp.
- Tóm tắt văn bản: Tạo bản tóm tắt ngắn gọn cho một văn bản dài hơn.
- Nhận dạng giọng nói: Chuyển đổi ngôn ngữ nói thành văn bản viết.
- Tạo văn bản: Tạo văn bản dựa trên lời nhắc hoặc ngữ cảnh nhất định.

- Hệ thống đối thoại: Xây dựng chatbot và tác nhân đàm thoại có thể tạo ra phản hồi phù hợp với thông tin đầu vào của người dùng.

7.4.7.3. Quy trình làm việc

Giai đoạn huấn luyện

1. Bộ mã hóa xử lý một chuỗi (ví dụ: một câu bằng tiếng Anh) và tạo ra các trạng thái ẩn.
2. Bộ giải mã tạo ra chuỗi đầu ra (ví dụ: bản dịch câu sang tiếng Pháp) bằng cách sử dụng các trạng thái ẩn này, có hoặc không có cơ chế chú ý hướng dẫn nó.

Giai đoạn suy luận

1. Với một chuỗi đầu vào mới, bộ mã hóa sẽ tạo ra các trạng thái ẩn.
2. Bộ giải mã sử dụng các trạng thái ẩn này để tạo ra chuỗi đầu ra theo từng bước, thường sử dụng các kỹ thuật như tìm kiếm chùm tia để tạo ra chuỗi có khả năng xảy ra nhất.

7.4.7.4. Những thách thức

- Xử lý các chuỗi dài: Các mô hình Seq2seq, đặc biệt là các mô hình có cấu trúc bộ mã hóa-giải mã đơn giản, có thể gặp khó khăn với các chuỗi rất dài. Cơ chế chú ý và kiến trúc tiên tiến hơn như Transformers giúp giải quyết vấn đề này.
- Căn chỉnh: Việc đảm bảo mô hình căn chỉnh chính xác các phần của chuỗi đầu vào và đầu ra có thể là một thách thức. Cơ chế chú ý giúp cải thiện sự liên kết bằng cách cho phép mô hình tập trung vào các phần có liên quan của đầu vào.

Các mô hình Seq2seq đã trở thành nền tảng trong xử lý ngôn ngữ tự nhiên và các tác vụ dựa trên trình tự khác, nhờ tính linh hoạt và hiệu quả của chúng trong việc xử lý các vấn đề khác nhau liên quan đến trình tự.

7.4.8. Mạng chuyển đổi

7.4.8.1. Giới thiệu

Với sự ra đời của cơ chế attention thì vào năm 2017 paper *Attention is all you need* đã giới thiệu một kiến trúc mới dành cho các bài toán NLP mà không có sự xuất hiện của các mạng nơ-ron hồi tiếp (RNN, LSTM,...) hay là mạng nơ-rơn tích chập (CNN) - đó là **Transformer**. Như đã giới thiệu ở bài

viết trước, trong các bài toán **seq2seq** các cấu trúc RNN hay LSTM đều có những hạn chế nhất định thì mô hình transformer đã khắc phục được những nhược điểm đó, khi giúp cho quá trình huấn luyện nhanh hơn và kết quả đạt được cũng tốt hơn. Hiện nay, transformer đã được phát triển thành nhiều biến thể khác nhau không chỉ phục vụ cho các bài toán seq2seq mà còn cho các mô hình ngôn ngữ.

Bảng dưới đây sẽ thể hiện sự khác biệt của transformer với các mô hình trước đó, khi transformer hoàn toàn hoạt động dựa trên cơ chế attention!

Bảng. Khác biệt giữa mạng chuyển đổi với mô hình khác

	seq2seq	seq2seq với attention	transformer
encoder	RNN/CNN	RNN/CNN	attention
decoder	RNN/CNN	RNN/CNN	attention
tương tác giữa encoder và decoder	vector	attention	attention

Cụ thể hơn một chút, thì ta sẽ hiểu tại sao transformer lại mang lại sự hiệu quả tốt hơn RNN. Đối với quá trình encode một câu thì RNNs sẽ cần một khoảng thời gian để đọc lần lượt từng từ trong câu, đối với 1 câu dài quá trình này có thể diễn ra khá lâu, như vậy độ phức tạp của encoder để xử lí một câu có độ dài N là O(N). Ngược lại, trong quá trình encode của transformer, các source tokens sẽ "quan sát" nhau (cơ chế self attention) và cố gắng hiểu nhau trong ngữ cảnh của câu. Và độ phức tạp của quá trình này chỉ là O(1).

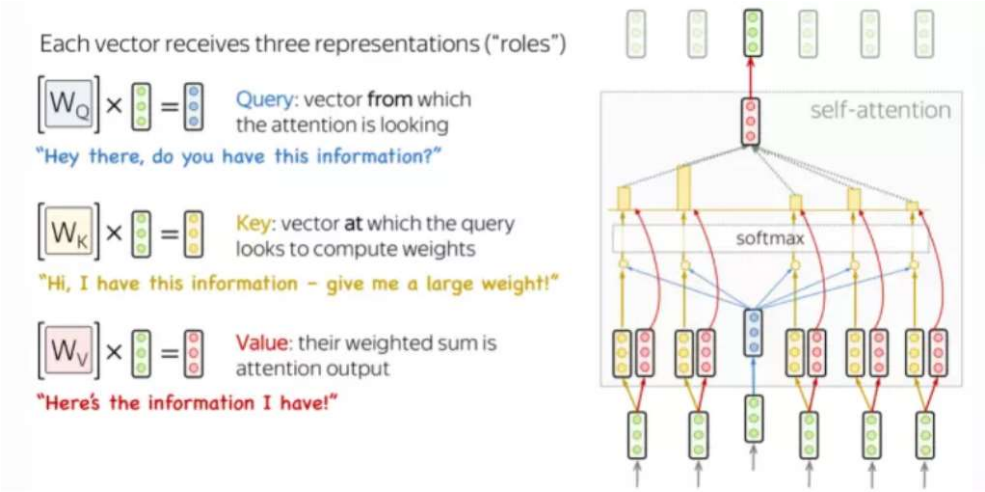
7.4.8.2. Cơ chế Self Attention

Có thể khẳng định rằng self -attention chính là thành phần quan trọng nhất của transformer. Sự khác biệt là, trong khi cơ chế attention sẽ tính toán dựa trên trạng thái của decoder ở time-step hiện tại và tất cả các trạng thái ẩn của encoder, Còn self-attention có thể hiểu là attention trong một câu, khi từng thành phần trong câu sẽ tương tác với nhau. Từng token sẽ "quan sát" các tokens còn lại trong, thu thập ngữ cảnh của câu và cập nhập vector biểu diễn.

Để xây dựng cơ chế self attention ta cần chú ý đến hoạt động của 3 vector biểu diễn cho mỗi từ lần lượt là:

- **Query**: hỏi thông tin
- **Key**: trả lời rằng nó có một số thông tin
- **Value**: trả về thông tin đó

Query được sử dụng khi một token "quan sát" những tokens còn lại, nó sẽ tìm kiếm thông tin xung quanh để hiểu được ngữ cảnh và mối quan hệ của nó với các tokens còn lại. Key sẽ phản hồi yêu cầu của Query và được sử dụng để tính trọng số attention. Cuối cùng, Value được sử dụng trọng số attention vừa rồi để tính ra vector đại diện (attention vector). Trong hình ba ma trận W, W_K và W_V chính là các hệ số mà mô hình cần huấn luyện.



Hình. Hệ số cần huấn luyện

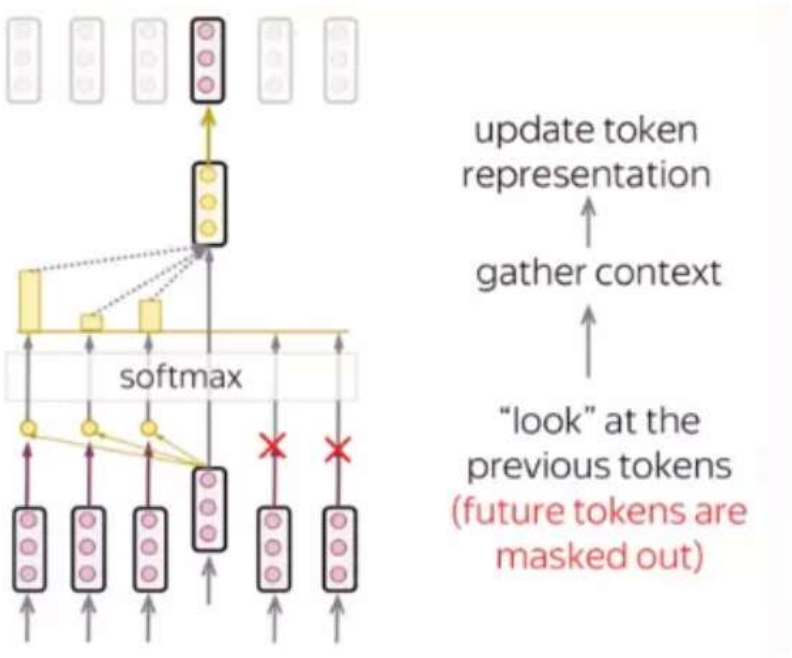
Biểu thức để tính attention vector như sau:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

Với d_k là số chiều của vector Key với mục đích tránh tràn luồng!

7.4.8.3. Cơ chế Masked Self Attention

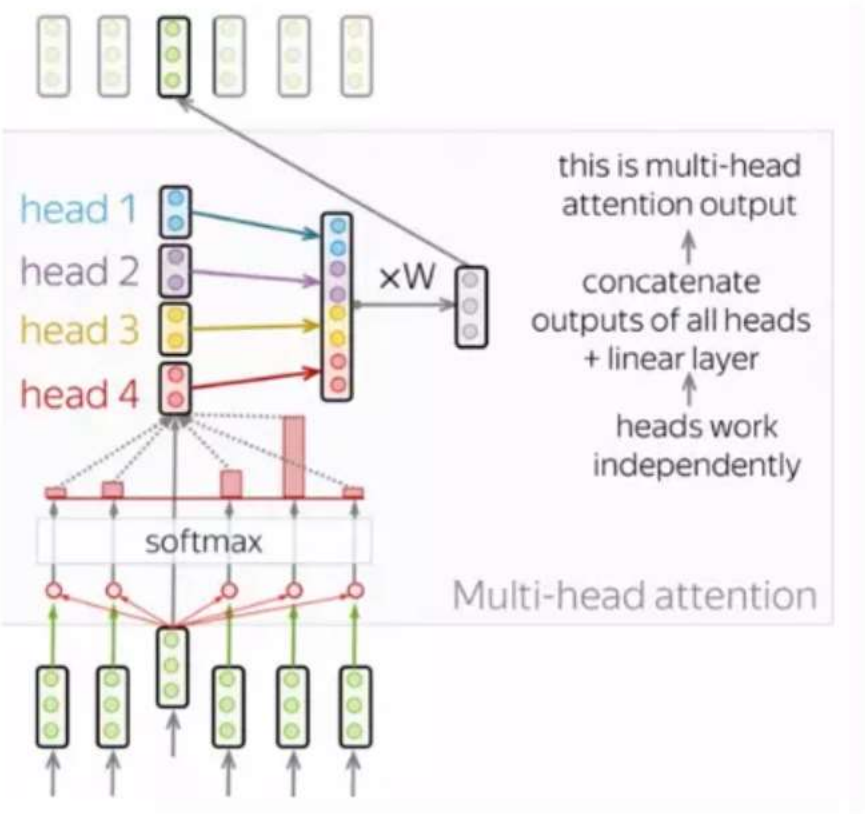
Đây là cơ chế được sử dụng cho decoder trong transformer, cụ thể nó thực hiện nhiệm vụ chỉ cho phép target token tại time-step hiện tại chỉ được phép dùng các tokens ở time-step trước đó. Về hoạt động nó cũng giống như đã giới thiệu ở trên, ngoại trừ việc nó không tính đến attention của những tokens trong tương lai.



Hình. Attention

7.4.8.4. Multi-Head Attention

Thông thường, để có thể hiểu được vai trò của một từ trong một câu ta cần hiểu được sự liên quan giữa từ đó và các thành phần còn lại trong câu. Điều này rất quan trọng trong quá trình xử lí câu đầu vào ở ngôn ngữ A và cả trong quá trình tạo ra câu ở ngôn ngữ B. Vì vậy, mô hình cần phải tập trung vào nhiều thứ khác nhau, cụ thể là thay vì chỉ có một cơ chế self attention như đã giới thiệu hay còn gọi là 1 "head" thì mô hình sẽ có nhiều "heads" mỗi head sẽ tập trung vào khía cạnh về sự liên quan giữa từ và các thành phần còn lại. Đó chính là multi-head attention.



Hình. Multi head attention

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_n)W$$

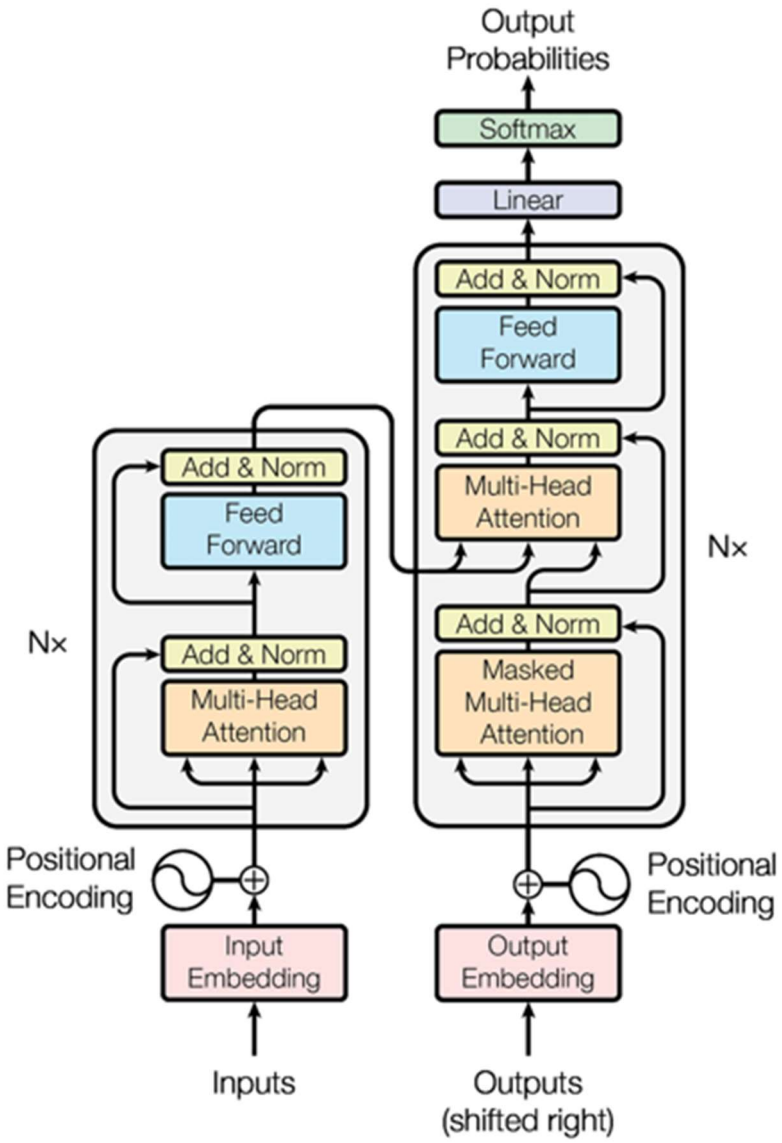
$$head_i = Attention(Q_i, K_i, V_i)$$

Khi triển khai, ta cần dựa vào query, key và value để tính toán cho từng head. Sau đó ta sẽ concat các ma trận thu được để thu được ma trận của multi-head attention. Để có đầu ra có cùng kích thước của đầu vào thì cần nhân với ma trận W.

7.4.8.5. Kiến trúc của Transformer

Tiếp theo ta sẽ đi tìm hiểu về các thành phần chính cấu tạo nên transformer. Đây là cấu trúc của transformer được giới thiệu trong bài báo *Attention is all you nee*d. Mô hình thực hiện chính xác những gì đã được giới thiệu ở trên. Ở bên trái là encoder, thông thường có Nx = 6 layers chồng lên nhau. Mỗi layer sẽ có multi-head attention như đã tìm hiểu và khối feed-forward. Ngoài ra còn các các kết nối residual giống như trong mạng Resnet. Ở bên phải là decoder, tương tự cũng có Nx = 6 layers chồng lên nhau. Kiến

trúc thì khá giống encoder những chỉ có thên khối masked multi-head attention ở vị trí đầu tiên. Ta sẽ tìm hiểu sâu hơn các thành phần trong transformer.



Hình. Thành phần trong mạng chuyển đổi

1. **Positional encoding**: Bởi vì transformer không có các mạng hồi tiếp hay mạng tích chập nên nó sẽ không biết được thứ tự của các token đầu vào. Vì vậy, cần phải có cách nào đó để cho mô hình biết được thông tin này. Đó chính là nhiệm vụ của positional encoding. Như vậy, sau bước nhúng từ (embedding

layers) để thu được các tokens thì ta sẽ cộng nó với các vector thể hiện vị trí của từ trong câu.

2. **Lớp Normalization**: Trong hình ảnh cấu trúc, có lớp "Add & Norm" thì từ Norm thể hiện cho lớp Normalization. Lớp này đơn giản là sẽ chuẩn hóa lại đầu ra của multi-head attention, mang lại hiệu quả cho việc nâng cao khả năng hội tụ.
3. **Kết nối Residual**[1]: Kết nối residual bản chất rất đơn giản: thêm đầu vào của một khối vào đầu ra của nó. Với kết nối này giúp mạng có thể chồng được nhiều layers. Như trên hình, kết nối residual sẽ được sử dụng sau các khối FFN và khối attention. Như trên hình từ "Add" trong "Add & Norm" sẽ thể hiện cho kết nối residual.
4. **Khối Feed-Forward**: Đây là khối cơ bản, sau khi thực hiện tính toán ở khối attention ở mỗi lớp thì khối tiếp theo là FFN. Có thể hiểu là cơ chế attention giúp thu thập thông tin từ những tokens đầu vào thì FFN là khối xử lí những thông tin đó.

Vừa rồi là một vài tìm hiểu về kiến trúc transformer. Transformer đã ra đời được khá lâu và không còn mới nhưng nó và những biến thể vẫn đang thể hiện một hiệu quả rất cao trong các bài toán NLP.

7.4.8.6. Mạng chuyển đổi trong Jupyter

Thư viện `transformers` của Hugging Face cung cấp các mô hình Transformer được đào tạo trước và một API đơn giản để tinh chỉnh và sử dụng chúng. Dưới đây là hướng dẫn từng bước về cách thiết lập và sử dụng mạng Transformer trong sổ ghi chép Jupyter.

- Bước 1. Nhập các thư viện

```
import torch
from transformers import AutoTokenizer,
AutoModelForSequenceClassification
```

- Bước 2. Tải mô hình huấn luyện trước và tách từ

```
# Load pre-trained model and tokenizer
model_name = 'bert-base-uncased'
tokenizer = AutoTokenizer.from_pretrained(model_name)
```

[1] Residual: dư, còn dư lại

```
model =
AutoModelForSequenceClassification.from_pretrained(model_
name)
```

- Bước 3. Nhập dữ liệu

```
texts = ["I love using Transformers!", "This is an
example sentence."]
# Tokenize input text
inputs = tokenizer(texts, padding=True, truncation=True,
return_tensors="pt")
```

- Bước 4. Dự báo

```
# Perform inference
with torch.no_grad():
 outputs = model(inputs)

# Extract logits (raw predictions)
logits = outputs.logits
print(logits)
```

- Bước 5. Giải thích kết quả

Có thể giải thích nhật ký đầu ra để hiểu dự đoán của mô hình. Ví dụ: nếu đang sử dụng mô hình để phân loại, có thể muốn áp dụng softmax để có xác suất.

```
import torch.nn.functional as F
# Convert logits to probabilities
probabilities = F.softmax(logits, dim=-1)
print(probabilities)
```

7.4.8.7. Sử dụng Thư viện `transformers` của TensorFlow và Hugging Face

Cũng có thể sử dụng TensorFlow với thư viện `transformers` của Hugging Face. Đây là một ví dụ tương tự với TensorFlow: Using TensorFlow and Hugging Face's `transformers` Library

- Bước 1. Nhập các thư viện

```
import tensorflow as tf
from transformers import
TFAutoModelForSequenceClassification, AutoTokenizer
```

- Bước 2. Tải mô hình huấn luyện trước và tách từ

```
# Load pre-trained model and tokenizer
model_name = 'bert-base-uncased'
tokenizer = AutoTokenizer.from_pretrained(model_name)
```

```
model =
TFAutoModelForSequenceClassification.from_pretrained(mode
l_name)
```

- Bước 3. Chuẩn bị dữ liệu vào

```
# Example input text
texts = ["I love using Transformers!", "This is an
example sentence."]
# Tokenize input text
inputs = tokenizer(texts, padding=True, truncation=True,
return_tensors="tf")
```

- Bước 4. Dự báo

```
# Perform inference
outputs = model(inputs)

# Extract logits (raw predictions)
logits = outputs.logits
print(logits)
```

- Bước 5. Giải thích kết quả

```
# Convert logits to probabilities
probabilities = tf.nn.softmax(logits, axis=-1)
print(probabilities)
```

Ví dụ về sử dụng máy biến áp với dữ liệu tùy chỉnh. Đây là một ví dụ đầy đủ hơn, bao gồm cách tinh chỉnh mô hình:

```
from transformers import Trainer, TrainingArguments
# Example data
train_texts = ["I love using Transformers!", "This is an
example sentence."]
train_labels = [1, 0] # Assuming binary classification

# Tokenize input text
train_encodings = tokenizer(train_texts, padding=True,
truncation=True, return_tensors="pt")

# Create dataset
class TextDataset(torch.utils.data.Dataset):
 def __init__(self, encodings, labels):
 self.encodings = encodings
 self.labels = labels

 def __getitem__(self, idx):
 item = {key: torch.tensor(val[idx]) for key, val in
self.encodings.items()}
 item['labels'] = torch.tensor(self.labels[idx])
 return item

 def __len__(self):
```

```
 return len(self.labels)

train_dataset = TextDataset(train_encodings,
train_labels)

# Define training arguments
training_args = TrainingArguments(
 per_device_train_batch_size=8,
 num_train_epochs=1,
 logging_dir='./logs',
)

# Define trainer
trainer = Trainer(
 model=model,
 args=training_args,
 train_dataset=train_dataset,
)

# Train model
```

Ví dụ này bao gồm các kiến thức cơ bản về tải, sử dụng và tinh chỉnh các mô hình Transformer trong sổ tay Jupyter với PyTorch và TensorFlow. Đối với các tác vụ phức tạp hơn hoặc các trường hợp sử dụng nâng cao, hãy tham khảo tài liệu của các thư viện hoặc chuyên ngành tương ứng.

7.4.8.8. Thí dụ mạng chuyển đổi trong Jupyter

Dịch câu tiếng Việt sang tiếng Anh

Để dịch văn bản từ tiếng Việt sang tiếng Anh bằng mô hình Transformer trong sổ tay Jupyter, có thể sử dụng mô hình MarianMT được hulu trước từ thư viện mạng chuyển đổi Hugging Face. MarianMT hỗ trợ nhiều tác vụ dịch thuật khác nhau, bao gồm dịch từ tiếng Việt sang tiếng Anh.

Dưới đây là hướng dẫn từng bước về cách thực hiện dịch từ tiếng Việt sang tiếng Anh bằng sổ tay Jupyter:

Hướng dẫn từng bước dịch từ tiếng Việt sang tiếng Anh.

1. Nhập thư viện

Nhập các thành phần cần thiết từ thư viện mạng chuyển đổi :

```
from transformers import MarianMTModel, MarianTokenizer
import torch
```

2. Tải Mô hình dịch và Tokenizer được huấn luyện trước

Để dịch từ Việt sang Anh, hãy sử dụng mô hình MarianMT được huấn luyện riêng cho cặp ngôn ngữ này. Hugging Face cung cấp mô hình cho nhiều cặp ngôn ngữ. Mô hình cần là Helsinki-NLP/opus-mt-vi-en:

```
model_name = 'Helsinki-NLP/opus-mt-vi-en'
tokenizer = MarianTokenizer.from_pretrained(model_name)
model = MarianMTModel.from_pretrained(model_name)
# cần cài đặt pip install sacremoses
```

3. Chuẩn bị văn bản đầu vào

Xác định văn bản tiếng Việt muốn dịch:

```
texts = [
 "Xin chào, tháng 7 năm 2024",
 "Tối nay có chung kết bóng đá EURO giữa đội tuyển Anh và
Tây Ban Nha",
 "Hẹn gặp lại nhé !"
]
```

4. Token hóa văn bản đầu vào

Mã hóa văn bản bằng cách sử dụng mã thông báo:

```
# Tokenize input text
inputs = tokenizer(texts, return_tensors="pt",
padding=True, truncation=True)
```

5. Thực hiện dịch thuật

Chuyển đầu vào được mã hóa thông qua mô hình để nhận bản dịch:

```
# Perform translation
with torch.no_grad():
 translated = model.generate(**inputs)

# Decode translated text
translated_texts = [tokenizer.decode(t,
skip_special_tokens=True) for t in translated]
print("Bản dịch sang tiếng Anh là :\n", translated_texts)

 Bản dịch sang tiếng Anh là :
 ['Hello, July 2024.', 'Tonight we have the same gay
soccer finals between the English and the Spanish team.',
"I'll see you later."]
```

Giải thích:

1. Helsinki-NLP/opus-mt-vi-en: Đây là mô hình MarianMT dịch từ tiếng Việt sang tiếng Anh.

2. tokenizer(texts, return_tensors="pt", đệm=True, truncation=True): Mã hóa văn bản tiếng Việt đầu vào, đảm bảo chúng được cắt bớt khi cần thiết, đồng thời chuyển đổi chúng thành các tensor PyTorch.

3. model.generate(**inputs): Tạo bản dịch tiếng Anh từ đầu vào tiếng Việt được mã hóa.

4. tokenizer.decode(t, Skip_special_tokens=True): Chuyển đổi ID mã thông báo đã tạo trở lại thành văn bản tiếng Anh có thể đọc được.

Thiết lập này cho phép dịch trực tiếp văn bản tiếng Việt sang tiếng Anh trong sổ ghi chép Jupyter bằng mô hình Transformer được huấn luyện trước.

Đối với các cặp ngôn ngữ khác hoặc các mô hình khác nhau, sẽ điều chỉnh tên mô hình và mã thông báo cho phù hợp.

7.5. Kết luận

Tài liệu tham khảo