به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

Trustworthy AI

تمرین شماره ۲

نام و نام خانوادگی	سارا رستمی
شماره دانشجویی	۸۱۰۱۰۰۳۵۵
تاریخ ارسال گزارش	

فهرست گزارش سوالات (لطفاً پس از تکمیل گزارش، این فهرست را به‌روز کنید.)

سوال ۱ – SHAP

الف)

۱.

به طور کلی روش‌های additive feature attribution یک مدل explanation دارند که یک تابع خطی از تعدادی متغیر باینری می‌باشد. این تابع طبق فرمول ۱ می‌باشد.

$$g(z') = \varphi_0 + \sum_{i=1}^{M} \varphi_i z'_i \qquad (۱)$$

به طوریکه $z' \in \{0,1\}^M$ و M تعداد featureهای ساده شده (simplified) و همچنین $\varphi_i \in \mathbb{R}$ می‌باشد.

در ادامه روش LIME (local interpretable model-agnostic explanations) را که جزو خانواده‌ی روش‌های additive feature attribution می‌باشد را توضیح می‌دهیم.

LIME

روش LIME پیش بینی های مدل به ازای یک نمونه را بر اساس تقریب محلی مدل حول یک پیش بینی معین تفسیر می کند. مدل توضیح خطی محلی که LIME استفاده می کند دقیقاً مطابق معادله ۱ می‌باشد، از این رو LIME یک روش additive feature attribution است. LIME ورودی های ساده شده x' را "ورودی های تفسیرپذیر" می‌نامد. نگاشت $x = h_x(x')$ یک بردار باینری از ورودی های تفسیرپذیر را به فضای ورودی اولیه تبدیل می کند. انواع مختلف نگاشت h_x برای input spaceهای مختلف استفاده می‌شود. برای ویژگی‌های bag of words، اگر ورودی ساده شده یک باشد، h_x بردار ۱ یا ۰ (حضور یا عدم حضور) را به تعداد کلمات اصلی تبدیل می کند، و اگر ورودی ساده شده صفر باشد، به صفر نگاشت می‌کند. برای تصاویر، h_x هر تصویر را به صورت مجموعه‌ای از ابرپیکسل‌ها در نظر می‌گیرد. سپس، ۱ را نگاشت میکند به اینکه مقدار ابرپیکسل اولیه دست نخورده(مقدار اولیه‌ش) باقی بماند و ۰ را نگاشت می‌کند به اینکه هر ابرپیکسل برابر شود با میانگین پیکسل‌های همسایه‌اش (در واقع این حالت نشان می‌دهد که ابرپیکسل حضور ندارد و باید مقداری به آن assign کرد).

برای پیدا کردن φ، LIME تابع هدف زیر را مینیمم می کند:

$$\xi = argmin_{g \in G} L(f, g, \pi_{x'}) + \Omega(g) \qquad (۲)$$

برای اینکه مدل توضیحی $g(z')$ تقریب خوبی از مدل اصلی ما (یعنی $f(h_x(z'))$) باشد، طبق فرمول ۲، از تابع Loss L روی مجموعه‌ای از نمونه‌های فضای ورودی ساده‌شده (simplified input space) که توسط kernel محلی $\pi_{x'}$ وزن‌دهی شده‌اند، استفاده می‌شود. تابع $\Omega(g)$ جلوی پیچیده‌شدن مدل g را می‌گیرد (به عبارتی پیچیده شدن مدل g را جریمه می‌کند). از آنجایی که g مطابق فرمول ۱، تعریف می‌شود، و تابع هزینه L از نوع squared loss می‌باشد، معادله ۲ را می‌توان با استفاده از روش penalized linear regression حل کرد.

از آنجایی که overviewی از روش LIME در سوال ۴ ارائه شد من روش DeepLIFT را هم در ادامه توضیح می‌دهم.

DeepLIFT

تکنیکی برای تفسیر شبکه‌های عمیق می‌باشد. DeepLIFT بر اساس مفهوم تخصیص امتیازهای مشارکت به هر ویژگی ورودی یا نورون است، که نشان دهنده‌ی میزان تأثیر آنها بر پیش بینی نهایی می‌باشد. این امتیازات مشارکت با مقایسه میزان activation یک ویژگی یا نورون با مقدار activation مرجع (baseline) محاسبه می‌شوند. این روش برای مقایسه با هر ویژگی ورودی یا نورون به یک مقدار پایه نیاز دارد. این baseline می تواند یک مقدار خنثی (به عنوان مثال، صفر) یا یک مقدار emprical (به عنوان مثال، مقدار متوسط در مجموعه داده آموزشی) باشد.

الگوریتم DeepLIFT با محاسبه تفاوت بین فعال‌سازی هر نورون و فعال‌سازی پایه متناظر با آن شروع می‌شود. این تفاوت نشان دهنده سهمی است که نورون در پیش بینی نهایی انجام می دهد. با انتشار این مشارکت‌ها به عقب (backward propagation) در شبکه، DeepLIFT امتیازهای مشارکت را برای هر نورون در هر لایه محاسبه می‌کند. هنگامی که امتیاز مشارکت برای هر نورون محاسبه شود، DeepLIFT این مشارکت ها را به ویژگی های ورودی نسبت می دهد. این امر با اعمال قانون زنجیره ای مشتقات برای تخصیص سهم به ویژگی های ورودی بر اساس تأثیر آنها بر فعال شدن هر نورون به دست می آید.

ویژگی local accuracy

این ویژگی بیان می‌کند که خروجی مدل توضیحی $g(x')$ زمانی که $x = h_x(x')$ باشد (یعنی زمانی که ورودی و ورودی ساده‌شده تقریبا برابر باشند) ، با خروجی خود مدل f(x) برابر است (طبق فرمول۳).

$$f(x) = g(x') = \varphi_0 + \sum_{i=1}^{M} \varphi_i x'_i \qquad (۳)$$

ویژگی missingness

در صورتی که ورودی‌های ساده‌شده (simplified inputs) نمایانگر حضور و عدم حضور ویژگی‌ها باشند، ویژگی missingness مجاب می‌کند که ویژگی‌هایی که در ورودی اصلی حضور ندارند، هیچ تاثیری در پیش‌بینی خروجی نداشته باشند (سهم آن‌ها در خروجی مدل باید ۰ باشد).

$$if\ x'_i = 0 \implies \varphi_i = 0$$

ویژگی consistency

این ویژگی بیان می‌کند اگر مدل به گونه‌ای تغییر کند که contribution یک ورودی ساده‌شده بدون توجه به سایر ورودی‌ها، افزایش یابد یا تغییری نکند، نباید سهم آن ورودی کاهش یابد (به طور مشابه اگر contribution این ویژگی کاهش یابد، نباید سهم آن ورودی افزایش یابد).

.۲

مشکل اصلی روش shapley values اصلی برای توضیح مدل این است که برای محاسبه‌ی shapley valueها، نیاز است که سهم هر ویژگی(coalition value) به ازای تمام جایگشت‌های ممکن از ویژگی‌ها محاسبه شود. که به این معناست که باید مدلمان را به این تعداد دفعات evaluate کنیم که کار پرهزینه‌ای است. به طور مثال برای مدلی با ۴ ویژگی، این تعداد برابر با ۶۴ خواهد بود در حالیکه برای مدلی با ۳۲ ویژگی به چیزی حدود ۱۷ میلیارد می‌رسد!

برای حل این مشکل Lundberg و Lee روش shapley kernel را ارائه کردند. این روش با در نظرگرفتن تعداد کمتری نمونه از جایگشت‌های ممکن، عمل می‌کند.

بنابراین کاری که ما انجام می‌دهیم این است که نمونه‌هایی از جایگشت‌های ویژگی‌های مختلف data point خاصی را که می‌خواهیم توضیح دهیم، از مدل عبور می‌دهیم. البته، بیشتر مدل‌های ML به ما اجازه نمی‌دهند تنها یک ویژگی را حذف کنیم، بنابراین کاری که ما انجام می‌دهیم این است که مجموعه داده‌های پس‌زمینه B را تعریف کنیم، همان مجموعه‌داده‌ای که مدل روی آن آموزش داده شده است. سپس ویژگی یا ویژگی‌های حذف شده خود را با مقادیری از مجموعه داده پس‌زمینه پر می‌کنیم، در حالی که ویژگی‌هایی را که در جایگشت گنجانده شده‌اند، روی مقادیر اصلی خود ثابت نگه می‌داریم. سپس میانگین خروجی مدل را بر روی تمام این data pointهای ساختگی جدید به عنوان خروجی مدل خود برای جایگشت ویژگی در نظر می گیریم، که آن را y bar می نامیم.

پس از محاسبه تعدادی نمونه به این روش، می‌توانیم آن را به عنوان یک رگرسیون خطی وزن‌دار فرموله کنیم و به هر ویژگی یک ضریب اختصاص دهیم. با یک وزن‌دهی مخصوص برای هر نمونه، بر اساس ترکیبی از تعداد کل ویژگی‌های مدل، تعداد coalitionهایی با تعداد ویژگی‌های مشابه با این نمونه خاص، و تعداد

ویژگی‌های گنجانده شده و حذف شده در این مدل جایگشت، اطمینان حاصل می کنیم که راه حل این رگرسیون خطی وزن‌دار به گونه ای است که ضرایب پیدا شده، معادل مقادیر Shapley باشد. این ایده وزن‌دهی اساس Shapley است و فرآیند رگرسیون خطی وزن‌دار به طور کلی Kernel SHAP است.

در حال حاضر، بسیاری از اشکال دیگر SHAP وجود دارد که در مقاله ارائه شده‌اند، مواردی که از مفروضات و بهینه‌سازی‌های خاص مدل برای سرعت بخشیدن به الگوریتم و فرآیند نمونه‌برداری استفاده می‌کنند، اما Kernel SHAP یکی از آنها است که universal بوده و می تواند برای هر نوع مدل یادگیری ماشینی اعمال شود.

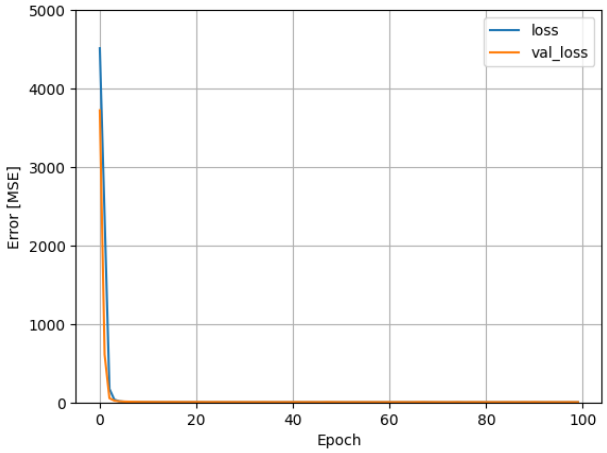
۳.

Kernel SHAP می‌تواند برای هر نوع مدلی، از جمله شبکه‌های عصبی عمیق، مورد استفاده قرار گیرد. درحالیکه DeepLIFT، به طور خاص برای شبکه‌های عصبی عمیق طراحی شده است، که سعی می‌کند عملکرد Kernel SHAP را بهبود ببخشد. DeepLIFT با تقریب مقدار Kernel SHAP کار می‌کند، اما فرض می‌کند که ویژگی‌های ورودی از هم مستقل هستند و شبکه عصبی عمیق خطی است. این بدان معناست که DeepLIFT بخش‌های غیر خطی شبکه عصبی را خطی می‌کند و این اجازه می‌دهد که تأثیر هر ویژگی بر خروجی مدل محاسبه شود. با این حال، DeepLIFT به طور هیوریستیک طراحی شده است، به این معنی که قواعد خطی کردن هر بخش از شبکه عصبی، بر اساس شهود تعیین شده‌اند و نه بر اساس یک اصل ریاضی. برای بهبود این مسأله، روش Deep SHAP توسعه داده‌شد که نقاط قوت Kernel SHAP و DeepLIFT را با یکدیگر ترکیب می‌کند. Deep SHAP مقادیر SHAP برای هر بخش شبکه عصبی محاسبه می‌کند و سپس آن‌ها را برای محاسبه مقادیر SHAP برای کل شبکه ترکیب می‌کند. این به این معناست که مقادیر کلی برای مدل به صورت کارآمد محاسبه می‌شود.

ب)

ابتدا داده‌ی Life Expectancy Data.csv را لود کرده و پیش‌پردازش‌های لازم (حذف سطرهای دارای مقادیر NaN، تغییر نام ستون‌ها به فرم استاندارد نام متغیرها) را روی آن انجام می‌دهیم. پس از آن با استفاده از تابع get_dummies() از کتابخانه‌ی pandas، دو متغیر categorical موجود در دیتاست (Country و Status) را به طور One-hot کد می‌کنیم. حال X و y را مشخص می‌کنیم. به طوریکه X شامل همه‌ی ستون‌ها و y حاوی ستون Life_expectancy می‌باشد. سپس ۱۰ درصد داده‌ها را به عنوان مجموعه تست و بقیه را به عنوان مجموعه آموزش در نظر می‌گیریم.

از آنجایی که مدل‌های ML به مقیاس داده‌ها حساس اند، نرمالایزر StandardScaler را روی داده‌های آموزش فیت کرده و transform می‌کنیم و داده‌های تست را با این نرمالایزر فیت شده، transform می‌کنیم. حال یک مدل MLP ساده برای رگرسیون با ۴ لایه طراحی می‌کنیم (تعداد نورون‌های هر لایه به ترتیب ۶۴، ۳۲، ۱۶ و نهایتاً ۱ می‌باشد). از کتابخانه tensorflow برای طراحی مدلمان استفاده می‌کنیم. از تابع Mean Squared Error به عنوان تابع loss و ADAM optimizer استفاده می کنیم. برای آموزش مدل ۲۰ درصد از داده‌های آموزش را به عنوان داده‌های validation در نظر می‌گیریم. نتیجه آموزش مدل را در شکل زیر مشاهده می‌کنید.



شکل ۱- نتیجه آموزش مدل **MLP** برای رگرسیون

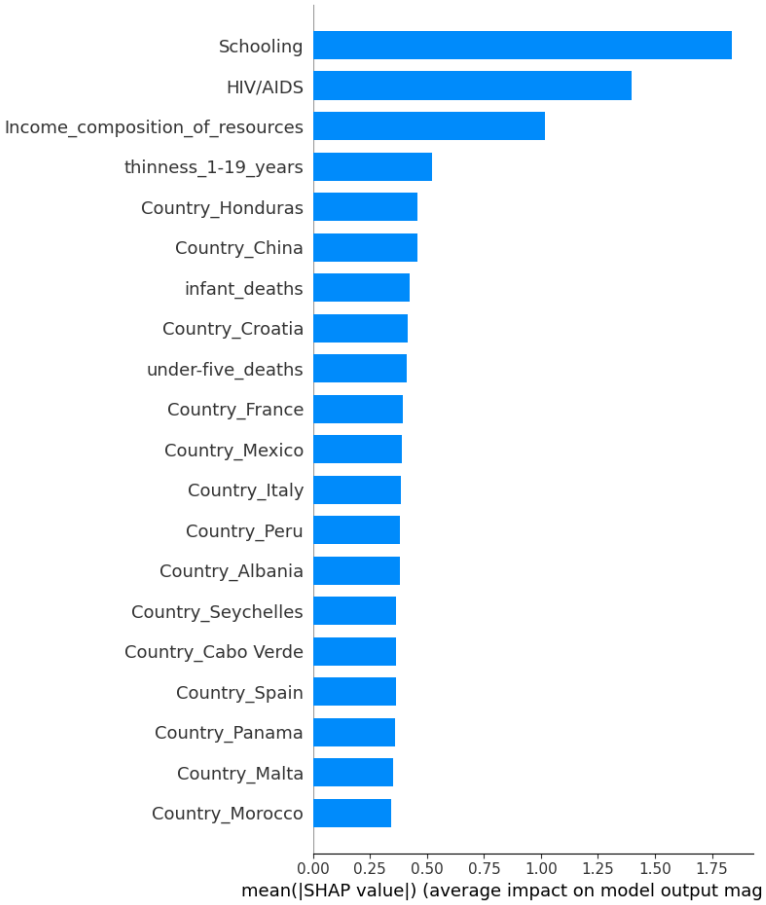
همانطور که در شکل بالا می‌بینید مدل به خوبی آموزش یافته. میزان loss و همچنین مقدار معیار R-squared مدل را روی داده‌های تست و آموزش در شکل زیر مشاهده می‌کنید.

```
47/47 [==============================] - 0s 2ms/step - loss: 2.3837
6/6 [==============================] - 0s 2ms/step - loss: 4.0585
Train performance -> R2^2: 0.9685210750534222
Test performance -> R2^2: 0.95577227341527
```

شکل ۲ - میزان **loss** و **R-squared** مدل **MLP** برای رگرسیون

همانطور که در شکل بالا می‌بینید مدل عملکرد خوبی هم روی داده‌های آموزش و هم روی داده‌های تست دارد. حال باید با استفاده از SHAP به میزان contribution هر ویژگی به پیش‌بینی مدل پی ببریم. برای این کار از کتابخانه‌ی shap استقاده می‌کنیم.

ابتدا روش Deep shap را امتحان می‌کنیم. مقادیر SHAP ویژگی‌ها را با استفاده از تابع DeepExplainer بدست می‌آوریم. سپس با استفاده از summary_plot() نمودار بارپلات ویژگی‌ها را به ترتیب میزان اهمیت آنها رسم می‌کنیم. این نمودار را در شکل زیر مشاهده می‌کنید.



شکل ۳- نمودار **summary_plot** نشان‌دهنده‌ی میزان اهمیت ویژگی‌ها با **Deep shap**

همانطور که در شکل بالا می‌بینید، ویژگی Schooling به عنوان مهمترین ویژگی در پیش‌بینی سن امید به زندگی یک کشور تشخیص داده‌شده. پس از ۴ ویژگی با اهمیت‌تر که در شکل بالا مشاهده می‌کنید، مقادیر one-hot شده‌ی ویژگی country را مشاهده می‌کنید. که نشان‌دهنده‌ی اهمیت چشمگیر ویژگی Country می‌باشد. اهمیت این ویژگی در تعیین میزان امید به زندگی افراد منطقی است چرا که هر کشور با توجه به شرایط سیاسی، اجتماعی و اقتصادی آن سطح رفاه خاص خود را برای مردمش فراهم می‌کند که مستقیماً در امید به زندگی آنها موثر است.

در force plot ، محور x مقادیر SHAP را نشان می دهد. تغییر رنگ از آبی به قرمز مقادیر shap ویژگی های کم اهمیت‌تر تا با اهمیت‌تر را نشان می‌دهد. ویژگی هایی که در این نمودار در کنار هم ظاهر شده‌اند، با هم interaction دارند.

حال دو کشور ارمنستان و ترکمنستان را به طور تصادفی از قاره آسیا انتخاب می‌کنیم و نمودار force plot را برای هر یک از آنها رسم می‌کنیم که در دو شکل زیر مشاهده می‌کنید. در این نمودار مقدار base value

نشان دهنده میانگین پیش بینی مدل در سراسر مجموعه داده است. طول میله ها یا نقاط نشان دهنده انحراف از مقدار پایه ناشی از هر ویژگی است.

شکل ۴- نمودار **force_plot** کشور ارمنستان با استفاده از **Deep shap**

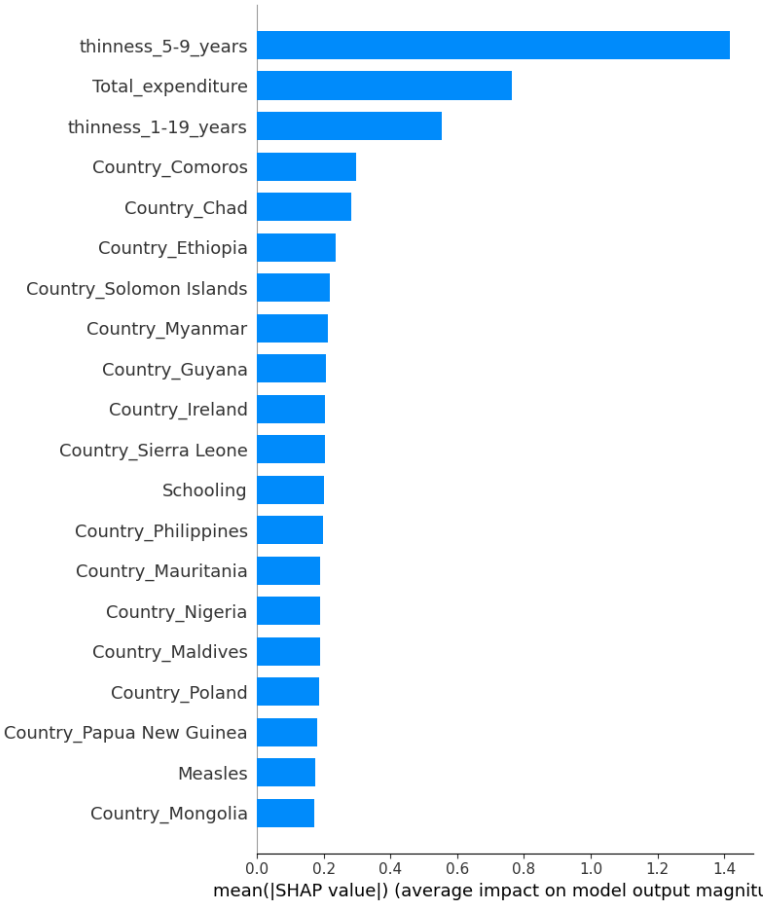
همانطور که در شکل بالا می‌بینید مهم‌ترین ویژگی برای این نمونه از کشور ارمنستان، ویژگی Country_Armenia می‌باشد که حاصل از one-hot کردن ویژگی country می‌باشد. که منطقی است. نمودار بالا نشان می‌دهد که ویژگی Country_Armenia به طور مثبت در پیش‌بینی سن امید زندگی برای این نمونه تاثیرگذار است . پس از آن به ترتیب ویژگی‌های Adult_Mortality و HIV/ADS مهم‌تر هستند.

شکل ۵- نمودار **force_plot** کشور ترکمنستان با استفاده از **Deep shap**

همانطور که در شکل بالا می‌بینید مهم‌ترین ویژگی برای این نمونه از کشور ترکمنستان، ویژگی Country_Turkmenistan می‌باشد که حاصل از one-hot کردن ویژگی country می‌باشد که با اهمیت بودن آن در پیش‌بینی خروجی منطقی است. پس از آن به ترتیب ویژگی‌های HIV/ADS و BMI مهم‌تر هستند.

از سه نمودار رسم شده با استفاده از Deep shap به میزان اهمیت ویژگی HIV/ADS و همچنین Country در سن امید به زندگی کشور، پی می‌بریم.

حال روش Kernel shap را امتحان می‌کنیم. مقادیر SHAP ویژگی‌ها را با استفاده از تابع KernelExplainer بدست می‌آوریم. سپس با استفاده از ()summary_plot نمودار بارپلات ویژگی‌ها را به ترتیب میزان اهمیت آنها رسم می‌کنیم. این نمودار را در شکل زیر مشاهده می‌کنید.



شکل ۶ - نمودار **summary_plot** نشان‌دهنده‌ی میزان اهمیت ویژگی‌ها با **kernel shap**

همانطور که در شکل بالا می‌بینید، ویژگی thinness_5-9_years به عنوان مهمترین ویژگی در پیش‌بینی سن امید به زندگی یک کشور تشخیص داده‌شده. همانند نمودار summary_plot حاصل از روش Deep shap، میزان اهمیت چشمگیر ویژگی Country مشخص می‌باشد.

با مقایسه‌ی نمودارهای این دو روش متوجه می‌شویم که تعدادی از ویژگی‌ها (همچون Country و Schooling) جزو پراهمیت‌ترین ویژگی‌ها از نظر هر دو explainer می‌باشد. از طرفی این دو Explainer در میزن اهمتی ویژگی‌های متعددی با هم اختلاف نظر دارند. این امر عجیب نیست.Kernel SHAP و Deep SHAP دو روش approximation متفاوت برای محاسبه موثر مقادیر Shapley هستند، بنابراین نباید انتظار داشت که آنها لزوماً موافق باشند.

سوال ۲ – Knowledge Distillation

۱.

انگیزه اصلی این مقاله، ایجاد مدلی بود که توضیح رفتار آن آسان باشد. درک نحوه کارکرد مدل‌های deep با بررسی نحوه عملکرد hidden unitهای آن کار دشوار و حتی غیرممکنی است. از طرفی، توضیح نحوه تصمیم‌گیری یک decision tree کار ساده‌ای‌ست. در این مقاله، یک راه جدید برای حل trade-off بین تعمیم و تفسیر پذیری، به نام Knowledge distillation، معرفی شده‌است. به جای تلاش برای درک چگونگی تصمیم گیری یک شبکه عصبی عمیق(DNN)، از شبکه عصبی عمیق برای آموزش درخت تصمیم استفاده شده که تابع ورودی-خروجی کشف شده توسط شبکه عصبی را تقلید می کند اما به روشی کاملاً متفاوت کار می کند. استفاده از soft decision (بر خلاف درخت تصمیم عادی که از hard decision باینری استفاده می‌کرد)، این قابلیت را به مدل می‌دهد که بیشتر بتواند پیچیدگی موجود در مدل‌های دیپ را capture کند.

از مزایای استفاده از این درخت تصمیم soft به جای DNNها به افزایش تفسیرپذیری، کاهش هزینه محاسباتی، بهبود تعمیم و کاهش نیاز به حافظه اشاره کرد.

درخت‌های تصمیم soft transparentتر و قابل درک‌تر از DNNها هستند، و توضیح نحوه رسیدن به یک تصمیم خاص را آسان‌تر می‌کنند. آنها همچنین به منابع محاسباتی کمتری نسبت به DNNها برای ارزیابی نیاز دارند. به‌علاوه، درخت‌های تصمیم soft می‌توانند بهتر از DNNها generalize دهند، به ویژه هنگامی که با مجموعه داده‌های آموزشی کوچک یا داده‌های شدیداً نویزی سروکار داریم.

همچنین، درخت‌های تصمیم soft حافظه کمتری نسبت به DNN مصرف می‌کنند، که امکان ذخیره‌سازی کارآمدتر و زمان‌های بارگذاری مدل سریع‌تر را فراهم می‌کند.

۲.

توانایی تعمیم بالای DNNها به خاطر بازنمایی‌های سلسله مراتبی توزیع‌شده‌ی (distributed hierarchical representations) آنهاست به این معنی که بازنمایی‌های تولیدشده در هر لایه به عنوان ورودی در لایه بعدی استفاده شده و تغییرات طی سلسله مراتبی روی بازنمایی‌ها اعمال می‌شود (به غیر از لایه اول که روی خود ورودی‌ها توابع فعالساز اعمال می‌شوند). از طرفی مدل درخت تصمیم بر مبنای hierarchical decision ها تصمیم‌گیری می‌کند. به این معنا که هر تصمیم ما را به تصمیم بعدی در نودی در level پایین تری از درخت می‌رساند.

به طور خاص این مدل درخت تصمیم soft، از شیوه‌ای به نام Hierarchical mixture of bigots (HMoB) استفاده می‌کند. در معماری HMoB، هر مدل MOE مجموعه ای از "experts" است، که مدل های individualی هستند که برای پیش بینی دقیق در زیر مجموعه های خاص فضای ورودی آموزش دیده اند. سپس خروجی‌های این expertها با استفاده از یک شبکه gating ترکیب می‌شوند تا پیش‌بینی نهایی را تولید کنند. expertها با استفاده از تکنیکی به نام Knowledge Distillation آموزش می بینند که شامل استفاده از خروجی های یک شبکه عصبی عمیق (DNN) از قبل آموزش دیده به عنوان "معلم" برای هدایت آموزش expertها است.

.۳

تابع هزینه cross entropy معمولاً در یادگیری عمیق برای کارهای طبقه بندی استفاده می شود. ای تابع هزینه، تفاوت بین توزیع احتمال پیش بینی شده و توزیع احتمال واقعی کلاس ها را اندازه گیری می کند. در درخت تصمیم soft ارائه شده در این مقاله، احتمالات پیش بینی شده با عبور دادن ورودی از طریق دنباله ای از قوانین تصمیم گیری (یک ساختار درختی) به جای شبکه عصبی محاسبه می شود.

تابع loss پیشنهادی برای درختان تصمیم soft ، از temperature scaling برای نرم کردن پیش بینی های درخت استفاده می کند. این شامل تقسیم logit ها (یعنی خروجی های قوانین تصمیم گیری) بر یک پارامتر دما است که منجر به توزیع احتمال گسترده تر می شود. سپس پیش بینی های نرم شده با برچسب های واقعی با استفاده از cross entropy loss مقایسه می شوند.

با استفاده temperature scaling، درخت تصمیم soft می‌تواند پیش‌بینی‌های پیوسته‌تری نسبت به درخت تصمیم استاندارد تولید کند. این در شرایطی مفید است که مرزهای تصمیم گیری بین کلاس ها مشخص نیست. علاوه بر این، پارامتر دما را می توان برای کنترل تعادل بین دقت و قابلیت تفسیر مدل تنظیم کرد. پارامتر دمای بالا مدل قابل تفسیرتری را تولید می کند، در حالی که پارامتر دمای پایین مدل دقیق تری تولید می کند.

.۴

برای جلوگیری از گیر افتادن در راه حل های ضعیف در طول آموزش، از یک term جریمه (penalty term) استفاده شده که هر نود داخلی را تشویق می کند تا از هر دو درخت فرعی چپ و راست به طور مساوی استفاده کند. بدون این جریمه، درخت تمایل داشت در plateau هایی گیر کند که در آن یک یا چند نود داخلی همیشه تقریباً تمام احتمالات را به یکی از درختان فرعی آن اختصاص می دادند و گرادیان logistic برای این تصمیم همیشه بسیار نزدیک به صفر بود.

به طور کلی regularization term به منظور جلوگیری از overfitting مدل استفاده می‌شود. Regularization term در اینجا متناسب با تعداد گره های درخت است و درخت را تشویق می کند تا گره های کمتری داشته باشد و در نتیجه پیچیدگی کمتری داشته باشد (پیچیدگی درخت را جریمه می‌کند).

سوال ۳ – D-RISE

a)

D-RISE یک روش برای تولید توضیحات بصری(visual explanation) برای object detectorها است. این روش Saliency mapهایی تولید می کند که نواحی از تصویر که بیشترین تأثیر را در object detection دارند، را نشان می‌دهد به طوریکه هم جنبه‌ی localization و هم classification را در object detection در نظر می‌گیرد. D-RISE نسبت به سایر روش‌های ارائه شده generalتر است و نیازی به دانش عملکرد داخلی مدل ندارد و این امر آن را برای object detectorهای مختلف (one-stage detectorهای مثل YOLOv3 و two-stage detctorهایی مثل Faster-RCNN) قابل استفاده می کند.

برخلاف روش‌هایی که الگوهای نوظهور را در وزن‌ها یا activationهای آموخته شده توضیح می‌دهند، تکنیک‌های attribution معمولاً شدیداً به طراحی مدل مرتبط هستند و بر تعدادی assumption در مورد معماری مدل تکیه می‌کنند. برای مثال Grad-CAM فرض می‌کند که هر feature map با مفهومی مرتبط است، و بنابراین، feature map ها را می‌توان با توجه به اهمیت مفهوم آنها برای category خروجی وزن‌دهی کرد. این مفروضات ممکن است برای مدل‌های تشخیص اشیا صدق نکنند، که منجر به شکست در تولید saliency mapهای با کیفیت می‌شود. علاوه بر این، object detectorها هم برای دسته‌بندی یک bounding box و هم برای مکان آن، به توضیح نیاز دارند. به این دلایل، استفاده مستقیم از تکنیک‌های attribution موجود در object detectorها غیرممکن است. D-RISE اولین روشی است که هر دو جنبه localization و طبقه‌بندی detection را توضیح می‌دهد. این مدل از مدل RISE الهام گرفته شده است و اثر mask کردن نواحی تصادفی در تصویر ورودی را بر خروجی پیش بینی شده برای تعیین میزان اهمیت آن ناحیه، اندازه گیری می کند.

b)

الگوریتم mask generation (که در RISE هم استفاده شده) به صورت زیر است:

گام ۱) تعداد N تا binary mask با سایز $h \times w$ (کوچکتر از سایز تصویر اصلی $H \times W$) انتخاب می‌کنیم (با setکردن هر پیکسل به طور مستقل از بقیه به ۱ با احتمال p و به ۰ با احتمال 1-p)

گام۲) با استفاده از bilinear interpolation ، همه‌ی maskها را به سایز $(h+1)C_H + (w+1)C_W$ Upsample می‌کنیم، به طوریکه $C_H \times C_W = \lfloor H/h \rfloor \times \lfloor W/w \rfloor$ برابر است با اندازه‌ی سلول در upsampled mask .

گام ۳) نواحی $H \times W$ را با offsetهای تصادفی یکنواخت در بازه‌ی (0,0) و (C_H, C_W) ، کراپ می‌کنیم.

با ترکیب این ماسک ها، الگوریتم یک saliency map تولید می کند که بااهمیت‌ترین نواحی شی را در تصویر ورودی هایلایت می کند.

(c

Similarity metric در این مقاله برای اندازه‌گیری شباهت بین بردار پیشنهادی (proposed vector) و بردار هدف (target vector) برای تشخیص شی استفاده می‌شود. هدف یافتن بردار پیشنهادی است که بیشتر شبیه به بردار هدف است، که می تواند به توضیح پیش بینی object detector کمک کند. متریک شباهت سه مؤلفه را ترکیب می کند: مجاورت مکانی (spatial proximity)، شباهت احتمال کلاس (class probability similarity) و شباهت نمره objectness (objectness score similarity). مجاورت مکانی با استفاده از Intersection over Union (IoU) اندازه گیری می شود، که یک معیار رایج برای ارزیابی عملکرد object detection است. شباهت احتمال کلاس با استفاده از cosine similarity احتمالات کلاس مرتبط با نواحی، اندازه گیری می شود. شباهت نمره objectness با گنجاندن معیاری از شباهت امتیازات objectness در metric، اندازه‌گیری می شود (اگر شبکه به صراحت آن را محاسبه کند). نویسندگان از این معیار تشابه استفاده کردند زیرا جنبه‌های مختلفی از شباهت بردار پیشنهادی با بردار هدف، از جمله نزدیکی مکانی، شباهت معنایی، و احتمال یک شی بودن بردار پیشنهاد را نشان می‌دهد. با گنجاندن هر سه مؤلفه در metric، نویسندگان توانستند saliency mapهای دقیق‌تر و قابل تفسیرتری برای object detectorها تولید کنند.

(d

روش پیشنهادی مبتنی بر ایجاد اختلال در تصویر ورودی و مشاهده تغییرات در خروجی object detector است. در حالی که این رویکرد می تواند در تولید saliency mapها موثر باشد، ممکن است تمام جنبه های فرآیند تصمیم گیری مدل را در بر نگیرد. یک راه پیشنهادی برای رفع این محدودیت می‌تواند

این باشد که مدل D-RISE را با روش‌های توضیحی دیگر مانند LIME و SHAP ترکیب کرد تا توضیحی کامل‌تر و دقیق‌تر از تصمیم object detector ارائه دهد.

همچنین، مکانیزم‌های توجه (attention) را می‌توان در مدل D-RISE گنجاند تا ویژگی‌های مرتبط‌تر را که object detector برای تصمیم‌گیری استفاده می‌کند، بهتر capture کند. این کار می تواند به تولید saliency mapهای تفسیرپذیرتر و دقیق تر کمک کند.

همچنین برای بهبود دقت و عملکرد مدل D-RISE می‌توان از فیدبک انسانی استفاده کرد. به طور مثال کاربران می توانند مناطقی از تصویر را که برای یک کار خاص مهم هستند هایلایت کنند و مدل می تواند saliency mapهای خود را بر این اساس refine و به روز کند.

e)

مدل‌های object detector چند خروجی تولید می کنند که حاوی class probabilities، اطلاعات localization مربوط به bounding box و امتیاز objectness می‌باشد. برای اعمال random mask به object detector، در این روش از امتیازهای localization و objectness در فرآیند تولید Saliency map استفاده می‌شود. با استفاده از یک object detector، یک تصویر، و یک bounding box برچسب‌گذاری شده، Saliency map تولید می‌شود که مناطقی که در پیش‌بینی bounding box توسط مدل مهم هستند را برجسته می‌کند.

تصویر اولی که به عنوان ورودی در نظر گرفتیم تصویر یک خرس می‌باشد که در شکل زیر مشاهده می‌کنید.

شکل ۷ - تصویر خرس ورودی شبکه

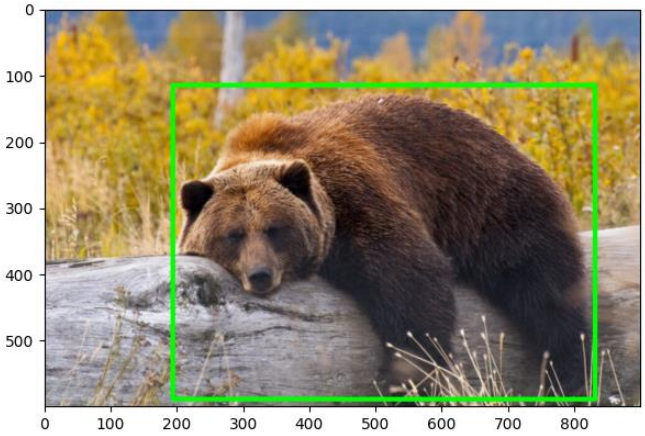
نتیجه‌ی پیش‌بینی مدل را در شکل زیر مشاهده می‌کنید. خروجی شامل مختصات bounding box پیشنهادی که شیء در آن قرار گرفته و دقت classifier برای تشخیص کلاس شیء (در اینجا خرس)

می‌باشد. عدد ۲۱ نشان‌دهنده‌ی شماره‌ی کلاس می‌باشد. مدل توانسته با دقت تقریبا کامل (۹۹ درصد) خرس را تشخیص دهد.

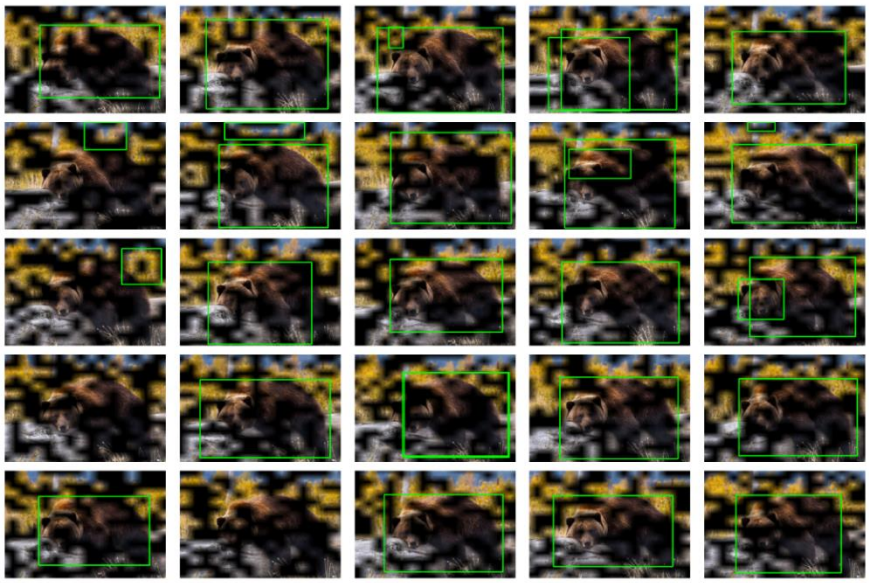
```
21 bear (193, 115, 830, 589) 0.99585044
```

شکل ۸ - خروجی شبکه به ازای تصویر خرس

در شکل زیر bounding box انتخاب شده از بین proposalها توسط مدل را مشاهده می‌کنید. همانطور که می‌بینید bounding box انتخابی به خوبی توانسته شکل خرس را در این تصویر احاطه کند.

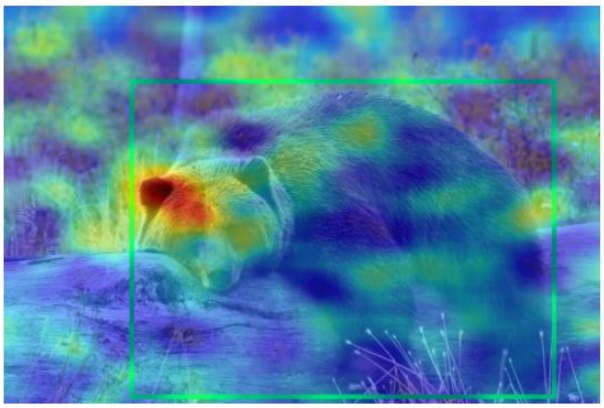


شکل ۹ - **bounding box** پیدا شده برای تصویر خرس



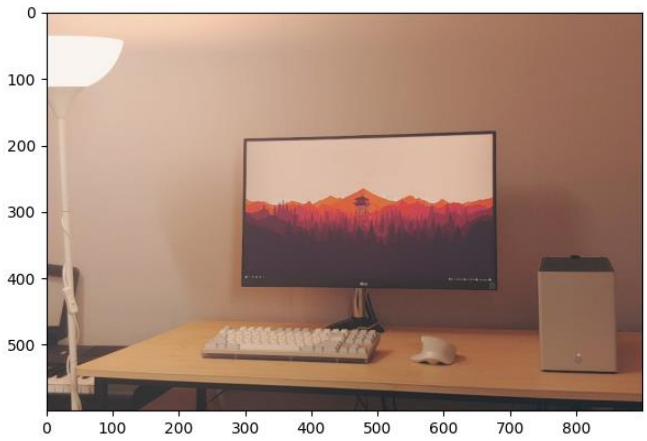
شکل ۱۰- **bounding box** های پیدا شده به ازای اضافه کردن نویزهای مختلف به تصویر

در شکل زیر Saliency Map حاصل را می بینید. رنگ قرمز به این معناست که مدل به نواحی تمرکز کرده و آبی به این معنی است که به اندازه کافی روی یک منطقه تمرکز نکرده است. همانطور که می‌بینید مدل نسبتاً خوب توانسته روی نواحی مهم، تمرکز کند (سر خرس مهم‌ترین ناحیه از نظر مدل بوده).



شکل ۱۱- **saliency map** تصویر خرس

تصویر دومی انتخابی شامل یک مانیتور و کیبورد و ماوس می‌باشد که در شکل زیر مشاهده می‌کنید.



شکل ۱۲- تصویر دوم به عنوان ورودی شبکه

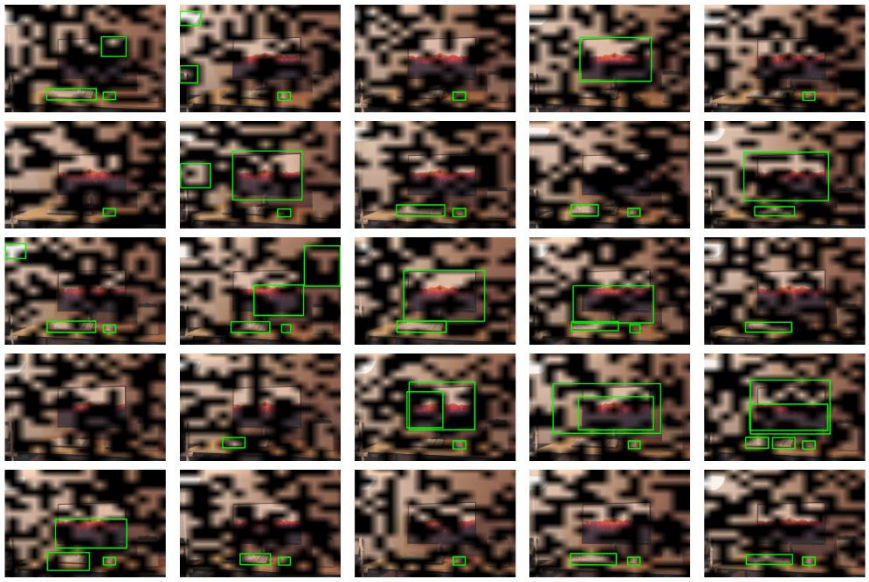
نتیجه‌ی پیش‌بینی مدل را برای این نمونه‌ی ورودی (تصویر دوم) در شکل زیر مشاهده می‌کنید. مدل توانسته با دقت خوبی اشیاء موجود در تصویر را تشخیص دهد (دقت شود که در کلاس‌های دیتاست کلاس مانیتور وجود نداشت و از این رو مدل مانیتور به عنوان تلویزیون تشخیص داده). مدل با احتمال ۹۹ درصد کیبورد را شناسایی کرده، با احتمال ۹۸ درصد ماوس را شناسایی کرده و با احتمال ۸۷ درصد مانیتور را تشخیص داده.

```
62 tv (289, 177, 682, 432) 0.8746213
64 mouse (550, 488, 619, 531) 0.9872649
66 keyboard (237, 467, 507, 530) 0.9955278
```

شکل ۱۳- خروجی شبکه به ازای تصویر دوم (مانیتور و کیبورد و ماوس)

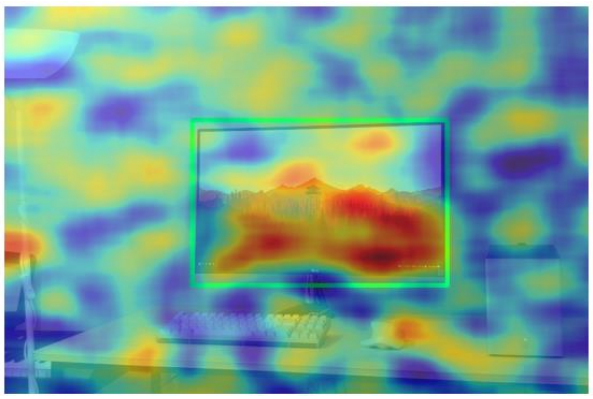
در شکل زیر bounding box های انتخاب شده از بین proposalها توسط مدل را مشاهده می‌کنید. همانطور که می‌بینید bounding box های انتخابی به خوبی توانستند اشیاء موجود در این تصویر را احاطه کنند.

شکل ۱۴- **bounding box** های پیدا شده برای تصویر دوم



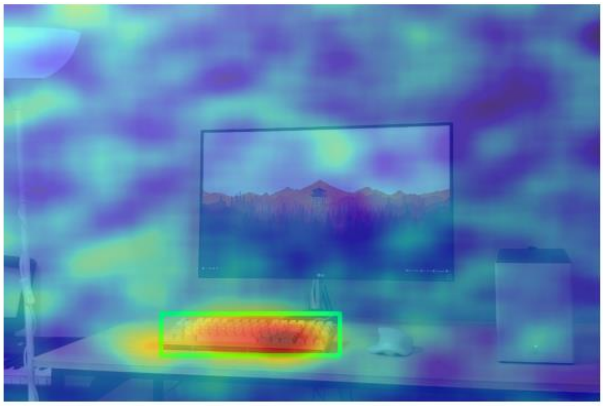
شکل ۱۵- **bounding box** های پیدا شده به ازای اضافه کردن نویزهای مختلف به تصویر

در شکل زیر Saliency Map مربوط به bounding box مانیتور را می بینید. مدل نسبتاً خوب توانسته روی نواحی مهم،که در اینجا صفحه مانیتور می‌باشد، تمرکز کند. البته مدل در این نمونه دارای خطاهایی هم می‌باشد. این خطاها در دقت تشخیص این شیء توسط مدل هم مشهود بود (چرا که مدل با دقت ۸۷ درصد مانیتور را به دلایل ذکر شده در بالا تلویزیون تشخیص داده). یک دلیل این خطاها این است که این شیء واقعا تلویزیون نیست بلکه مانیتوری است که روی یک میز کامپیوتر با تجهیزات کامپیوتری قرار گرفته.



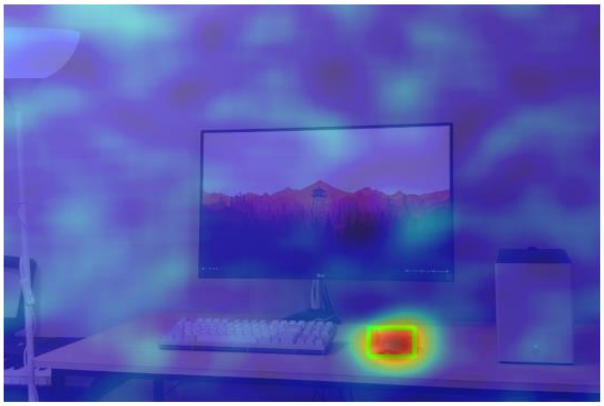
شکل ۱۶- **saliency map** تصویر دوم (برای مانیتور)

در شکل زیر Saliency Map مربوط به bounding box کیبورد را می بینید. مدل خیلی خوب توانسته روی نواحی مهم تصویر، برای تشخیص کیبورد تمرکز کند.



شکل ۱۷- **saliency map** تصویر دوم (برای کیبورد)

Saliency Map مربوط به bounding box ماوس را در شکل زیر مشاهده می‌کنید. برای این شیء هم مدل توانسته بسیار دقیق عمل کند و روی نواحی مهم تصویر تمرکز کند.



شکل ۱۸- **saliency map** تصویر دوم (برای ماوس)

تصویر سوم انتخابی شامل یک کیک و تعدادی دونات روی آن کیک می‌باشد که در شکل زیر مشاهده می‌کنید.

شکل ۱۹- تصویر سوم به عنوان ورودی شبکه

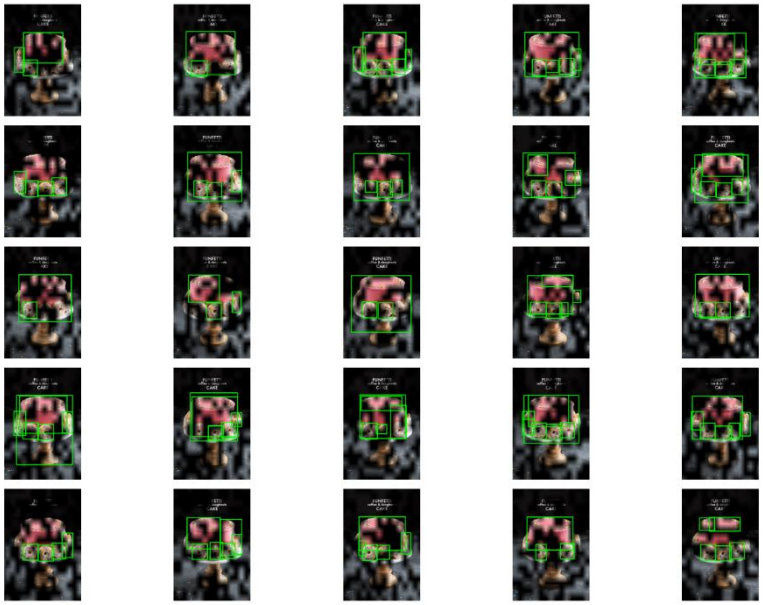
نتیجه‌ی پیش‌بینی مدل را برای این نمونه‌ی ورودی (تصویر سوم) در شکل زیر مشاهده می‌کنید. مدل توانسته با دقت خوبی اشیاء موجود در تصویر را تشخیص دهد. مدل ۴ تا دونات را به ترتیب با احتمال‌های ۹۷، ۹۷ ، ۹۴ و ۵۵ درصد در تصویر تشخیص داده و کیک را با احتمال ۷۸ درصد و میز نهارخوری را با احتمال ۸۴ درصد تشخیص داده که تنها شیء به اشتباه تشخیص داده شده توسط مدل بوده است.

```
54 donut (147, 432, 262, 557) 0.9797856
54 donut (263, 440, 384, 562) 0.9793559
54 donut (372, 427, 478, 544) 0.9474226
54 donut (125, 226, 479, 480) 0.55363756
55 cake (83, 227, 521, 608) 0.7896607
60 dining table (0, 306, 600, 845) 0.84187955
```

شکل ۲۰- خروجی شبکه به ازای تصویر سوم

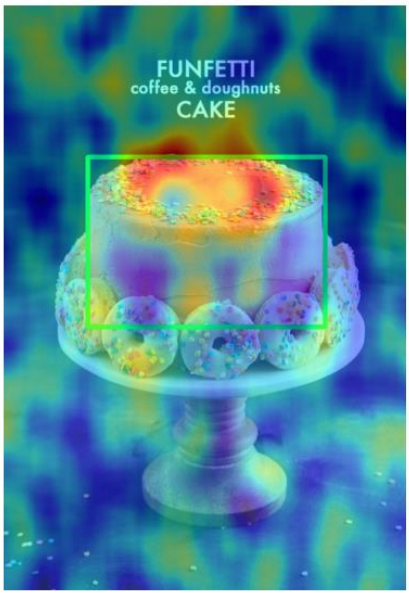
در شکل زیر bounding box های انتخاب شده از بین proposalها توسط مدل را مشاهده می‌کنید. همانطور که می‌بینید bounding box های انتخابی به خوبی توانستند اشیاء موجود در این تصویر را احاطه کنند (به غیر از bounding box بزرگ که میز نهارخوری را تشخیص داده که البته با توجه به context تصویر، پیش‌بینی چندان پرت و بی‌ربطی نبوده است.

شکل ۲۱- **bounding box** های پیدا شده برای تصویر سوم



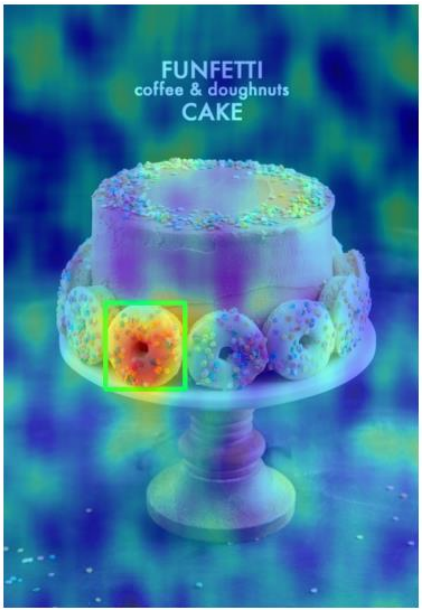
شکل ۲۲- **bounding box** های پیدا شده به ازای اضافه کردن نویزهای مختلف به تصویر

در شکل زیر Saliency Map مربوط به bounding box کیک را می بینید. مدل نسبتاً خوب توانسته روی نواحی مهم، بدنه‌ی کیک، تمرکز کند. البته مدل در این نمونه دارای خطاهایی هم می‌باشد. این خطاها در دقت تشخیص این شیء توسط مدل هم مشهود بود (چرا که مدل با دقت ۷۸ درصد کیک را تشخیص داده).



شکل ۲۳- **saliency map** تصویر سوم (برای کیک)

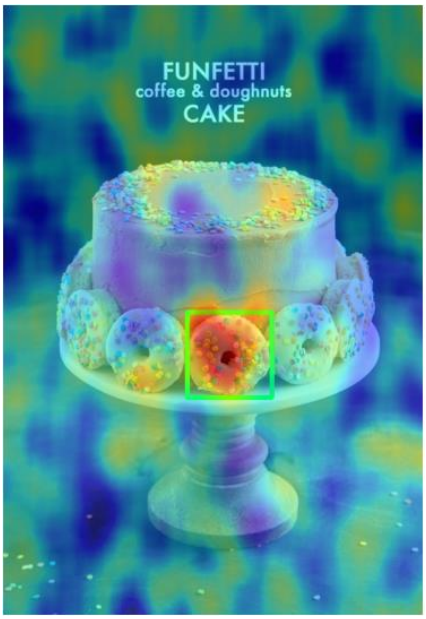
در شکل زیر Saliency Map مربوط به bounding box یکی از دونات‌ها را می بینید. مدل خوب توانسته روی نواحی مهم، دونات ، تمرکز کند (میزان confidence مدل برای پیش‌بینی این دونات ۹۷ درصد بود که با saliency map شکل پایین نیز همخوانی دارد).



شکل ۲۴- **saliency map** تصویر سوم (برای یکی از دونات‌ها)

در شکل زیر Saliency Map مربوط به bounding box یکی دیگر از دونات‌ها را می بینید. مدل همانند دونات قبلی خوب توانسته روی نواحی مهم، تمرکز کند (میزان confidence مدل برای پیش‌بینی این

دونات ۹۷ درصد بود که با saliency map شکل پایین نیز همخوانی دارد. از این نظر که مپ با دقت خوبی نواحی مهم را شناسایی کرده).



شکل ۲۵- **saliency map** تصویر سوم (برای یکی از دونات‌ها)

در کل مدل به خوبی می‌تواند bounding box های مربوط به اشیاء را پیدا کند و نسبتاً خوب می‌تواند نواحی مهم تصویر را برای شناسایی هر شیء شناسایی کند.

سوال ۴ – LIME

a)

مدل آموزش یافته‌ی MobileNetV2 آموزش یافته با دیتاست imagenet را توسط فریمورک tensorflow لود می‌کنیم (مدل MobileNetV2 را از tensorflow.keras.applications.mobilenet_v2 import می‌کنیم و پارامتر weights آن را برابر با ’imagenet‘ قرار می‌دهیم).

(b

تصویر انتخابی تصویر یک حلزون (کلاس snail) می‌باشد که آن را در شکل زیر مشاهده می‌کنید.

شکل ۲۶- تصویر حلزون به عنوان ورودی **test** برای شبکه **MobileNetV2**

برای دادن این تصویر به شبکه pretrainedمان ابتدا لازم است پیش‌پردازش‌های لازم را روی آن اعمال کنیم تا برای ورود به شبکه مناسب باشد. برای این کار به شیوه نشان داده شده در تکه کد زیر عمل می‌کنیم.

```
img = skimage.io.imread(url)
img = skimage.transform.resize(img, (224,224))
img = (img - 0.5)*2
img = np.expand_dims(img, axis=0)
```

شکل ۲۷- پیش‌پردازش تصاویر ورودی به شبکه از پیش آموزش یافته **MobileNetV2**

سپس خروجی شبکه را به ازای ورودی پیش‌پردازش شده بدست می‌آوریم. و با استفاده از تابع decode_predictions()، ۵ پیش‌بینی محتمل‌تر شبکه را نمایش می‌دهیم که نتیجه آن را در شکل زیر مشاهده می‌کنید.

```
- snail(94.56%)
- slug(0.17%)
- isopod(0.14%)
- hermit_crab(0.10%)
- conch(0.08%)
```

شکل ۲۸- پنج دسته با بالاترین احتمال طبق خروجی شبکه ازپیش آموزش یافته **MobileNetV2** به ازای تصویر ورودی حلزون

همانطور که در این شکل مشاهده می‌کنید، شبکه به درستی موفق شده تصویر را در کلاس snail طبقه‌بندی کند و این احتمال با اختلاف زیاد از احتمال‌های ۴ کلاس بعدی بیشتر است. به عبارتی شبکه با اطمینان بالا تصمیم خود را گرفته است.

(c

در این مرحله یک توضیح دهنده LIME ایجاد می‌کنیم و توضیح را روی مدل ازپیش‌آموزش یافته‌ی MobileNetV2 و تصویر اجرا می‌کنیم. ماژول مربوط به کار با تصویر از کتابخانه lime را به این صورت زیر تعریف کردیم:

$$explainer = lime\_image.LimeImageExplainer()$$

$$explanation = explainer.explain\_instance(images\_inc\_im[0].astype('double'), model.predict, hide\_color = 0, top\_labels = 5, num\_samples = 1000)$$

در instance explanation، num_features=5 به این معنی است که می‌خواهیم ۵ ویژگی برتر (منطقه‌های segment شده) را استخراج کنیم که بیشترین سهم را در پیش‌بینی حیوان دارند (حلزون)، به عبارت دیگر، مدل ما به شدت به ۵ ناحیه تقسیم‌بندی شده بستگی دارد که snail را در تصویر تشخیص می‌دهند.

(d

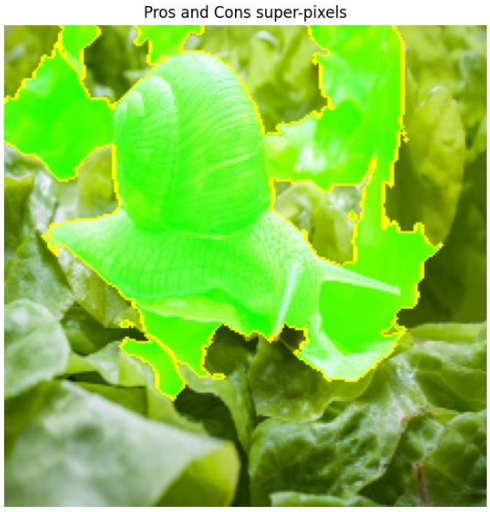
مرزهای (boundaries) بدست آمده توسط model interpreter (یعنی LIME) را روی تصویر وروی رسم می‌کنیم که آن را در شکل زیر مشاهده می‌کنید.

شکل ۲۹- مرزهای رسم شده بر روی نواحی **Super-pixel** برای تصویر حلزون

همانطور که در شکل بالا مشاهده می‌کنید super-pixelهایی که برای تعیین مرزها انتخاب شدند شکل حلزون را تقریباً در بر می‌گیرند. این بدان معناست که مدل ما تصویر ما را به دلیل این بخش های سوپرپیکسل به عنوان حلزون طبقه بندی می کند.

e)

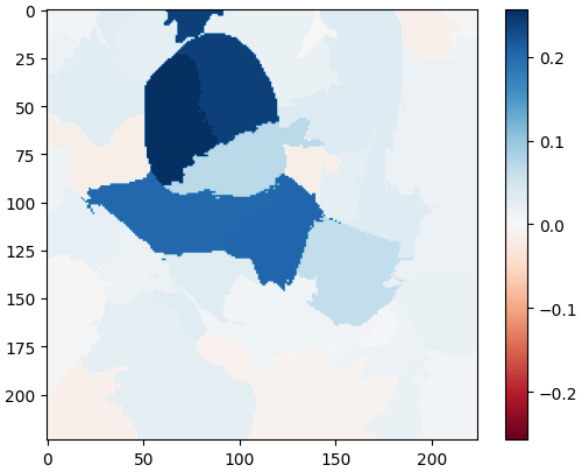
حال نواحی که در تشخیص تصویر ورودی به شبکه کمک بیشتری کردند (طبق LIME اهمیت بیشتری داشتند) را روی تصویر ورودی مشخص می‌کنیم. در تصویر زیر، ناحیه سوپرپیکسل‌هایی که به رنگ سبز رنگ شده‌اند، آن‌هایی هستند که احتمال تعلق تصویر ما به کلاس حلزون را افزایش می‌دهند، در حالی که سوپرپیکسل‌های رنگ‌شده با رنگ قرمز آن‌هایی هستند که احتمال را کاهش می‌دهند. در تصویر زیر سوپرپیکسل‌های قرمزی مشاهده نمی‌شود چرا که تعداد sampleهایی که با perturb کردن تصویر اصلی تولید کردیم ۱۰۰۰تاست که تعداد نسبتاً زیادی است. در ازای تعداد کمتری sample (مثلا ۱۰۰تا که تست کردیم)، نواحی قرمز مشخص شده بودند.



شکل ۳۰ - مشخص کردن **Pros** و **Cons** برای تصویر حلزون

f)

در شکل زیر heatmap مربوط به نواحی موثر و میزان contribution آنها را مطابق با رنگشان مشاهده می‌کنید. میزان contribution در واقع ضرایب featureها در رگرسیون خطی surrogate (استفاده شده در LIME) است.



شکل ۳۱- **heatmap** حاصل از **LIME** برای تصویر حلزون

همانطورکه در شکل بالا می‌بینید، نواحی آبی پررنگ که شامل بدن و صدف حلزون می‌باشند، بیشترین contribution مثبت را برای طبقه‌بندی حلزون داشتند. این heatmap با نواحی Pros و Cons مشخص شده مطابقت دارد.

(g

تصویر انتخابی بعدی (تصویر دوم) شامل سه تا از categoryهای موجود در دیتاست imagenet می‌باشد. این تصویر را در شکل زیر مشاهده می‌کنید.

شکل ۳۲- تصویر دوم به عنوان ورودی **test** برای شبکه **MobileNetV2**

همانطور که می‌بینید این تصویر شامل پیتزا، شراب قرمز و بطری شراب می‌باشد. در شکل زیر ۵ پیش‌بینی محتمل‌تر شبکه را به ازای این تصویر می‌بینیم.

```
- pizza(35.83%)
- red_wine(14.68%)
- wine_bottle(4.82%)
- frying_pan(3.78%)
- potpie(2.34%)
```

شکل ۳۳- پنج دسته با بالاترین احتمال طبق خروجی شبکه ازپیش آموزش یافته **MobileNetV2** به ازای تصویر ورودی دوم

همانطور که در شکل بالا می‌بینید، ۳ پیش‌بینی اول شبکه به ترتیب پیتزا، شراب قرمز و بطری شراب می‌باشد. پس شبکه به خوبی توانسته اشیاء را شناسایی کند. حال مدل MobileNetV2 و این تصویر را به LIME explainer می‌دهیم و پارامترهای آن را همانند بخش قبل (همانطور که برای تصویر حلزون انجام دادیم) مقداردهی می‌کنیم. نتیجه‌ی مرزکشی اطراف ۵ تا سگمنت مهمتر را در شکل زیر مشاهده می‌کنید.

شکل ۳۴- مرزهای رسم شده بر روی نواحی **Super-pixel** برای تصویر دوم

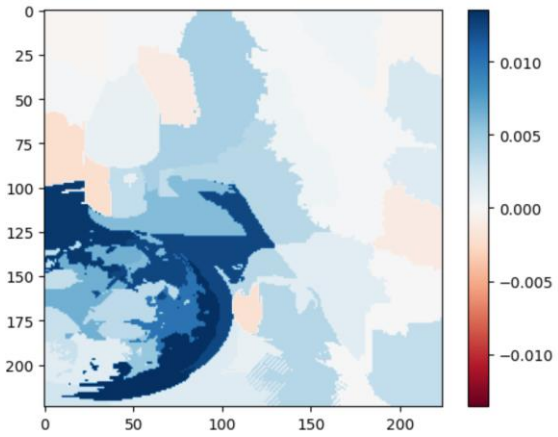
طبق شکل بالا، مهمترین سوپرپیکسل‌ها مربوط به نواحی حول و حوش پیتزا می باشند که این مشاهده با نتیجه ی بدست آمده از مدلمان (۵ تا خروجی محتمل‌تر مدل) همخوانی دارد.

سوپرپیکسل‌هایی که contribution مثبت و منفی داشتند را در شکل زیر مشاهده می‌کنید.

شکل ۳۵- مشخص کردن **Pros** و **Cons** برای تصویر دوم

همانطور که در شکل بالا مشخص است، مدل بیشتر در انتخاب نواحی مربوط به پیتزا درست عمل کرده و پس از آن قسمت‌هایی از بطری شراب و لیوان شراب را نیز انتخاب کرده‌است. این نتایج با نتایج بدست آمده در مرزکشی‌های دور سگمنت‌های مثبت و همچنین ۵ خروجی محتمل مدل همخوانی دارد.

در شکل زیر heatmap مربوط به نواحی موثر و میزان contribution آنها را مطابق با رنگشان مشاهده می‌کنید.



شکل ۳۶- **heatmap** حاصل از **LIME** برای تصویر دوم

همانطورکه در شکل بالا می‌بینید، نواحی آبی پررنگ که فرم دایره‌ای شکل پیتزا را capture می‌کند، بیشترین contribution مثبت را برای طبقه‌بندی اشیاء داشتند. در واقع بیشترین contribution را برای تشخیص پیتزا داشتند که طبق ۵ پیش‌بینی محتمل‌تر مدل انتظار می‌رفت (از آنجایی که پیتزا را با بیشترین احتمال پیش‌بینی می‌کرد). این heatmap با نواحی Pros و Cons مشخص شده نیز مطابقت دارد.

تصویر انتخابی بعدی (تصویر سوم) شامل دو تا از categoryهای موجود در دیتاست imagenet می‌باشد. این تصویر را در شکل زیر مشاهده می‌کنید.

شکل ۳۷- تصویر سوم به عنوان ورودی **test** برای شبکه **MobileNetV2**

همانطور که می‌بینید این تصویر شامل خرس عروسکی (teddy bear) و عینک آفتابی می‌باشد. در شکل زیر ۵ پیش‌بینی محتمل‌تر شبکه را به ازای این تصویر می‌بینیم.

```
- teddy(83.26%)
- sunglass(3.69%)
- sunglasses(1.28%)
- tennis_ball(0.38%)
- Band_Aid(0.24%)
```

شکل ۳۸- پنج دسته با بالاترین احتمال طبق خروجی شبکه ازپیش آموزش یافته **MobileNetV2** به ازای تصویر ورودی سوم

همانطور که در شکل بالا می‌بینید، ۲ پیش‌بینی اول شبکه به ترتیب خرس عروسکی و عینک آقتابی می‌باشد. پس شبکه به خوبی توانسته اشیاء را شناسایی کند. حال مدل MobileNetV2 و این تصویر را به LIME explainer می‌دهیم و پارامترهای آن را همانند بخش قبل (همانطور که برای تصویر حلزون انجام دادیم) مقداردهی می‌کنیم. نتیجه‌ی مرزکشی اطراف ۵ تا سگمنت مهمتر را در شکل زیر مشاهده می‌کنید.

شکل ۳۹- مرزهای رسم شده بر روی نواحی **Super-pixel** برای تصویر دوم

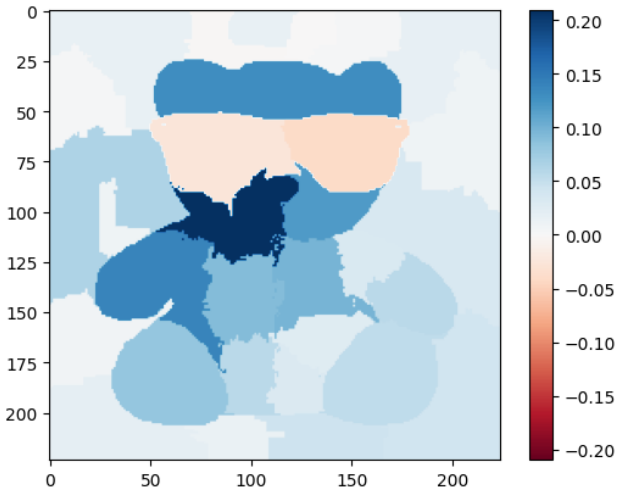
طبق شکل بالا، مهمترین سوپرپیکسل‌ها مربوط به نواحی حول و حوش بدن خرس و اطراف عینک آفتابی و همچنین گوش‌های خرس می باشند که این مشاهده با نتیجه ی بدست آمده از مدلمان (۵ تا خروجی محتمل‌تر مدل) تا حدودی همخوانی دارد.

سوپرپیکسل‌هایی که contribution مثبت و منفی داشتند را در شکل زیر مشاهده می‌کنید.

شکل ۴۰- مشخص کردن **Pros** و **Cons** برای تصویر سوم

همانطور که در شکل بالا مشخص است، مدل بیشتر در انتخاب نواحی مربوط به بدن خرس درست عمل کرده‌است. این نتایج با نتایج بدست آمده در مرزکشی‌های دور سگمنت‌های مثبت و همچنین ۵ خروجی محتمل مدل همخوانی دارد.

در شکل زیر heatmap مربوط به نواحی موثر و میزان contribution آنها را مطابق با رنگشان مشاهده می‌کنید.



شکل ۴۱- **heatmap** حاصل از **LIME** برای تصویر سوم

همانطورکه در شکل بالا می‌بینید، نواحی آبی پررنگ‌تر که تا حد زیادی outline خرس عروسکی را capture می‌کند و بیشترین contribution مثبت را برای طبقه‌بندی اشیاء داشتند. در واقع بیشترین contribution را برای تشخیص خرس عروسکی داشتند که طبق ۵ پیش‌بینی محتمل‌تر مدل انتظار می‌رفت (از آنجایی که خرس را با بیشترین احتمال پیش‌بینی می‌کرد). از طرفی سگمنت‌های مربوط به عینک، بیشترین contribution منفی را در تشخیص خرس داشتند و به عبارتی شبکه به خوبی توانایی تمیز کردن بین خرس عروسکی و عینک آفتابی را دارد. این heatmap با نواحی Pros و Cons مشخص شده نیز مطابقت دارد.