دانشگاه صنعتی شریف

دانشکده مهندسی برق

گزارش پروژه درس یادگیری عمیق

نویسندگان:

احمدرضا رحیم‌زارع - ۹۹۲۰۶۰۳۳

امیرحسین اسدیان – ۹۶۱۰۱۱۸۷

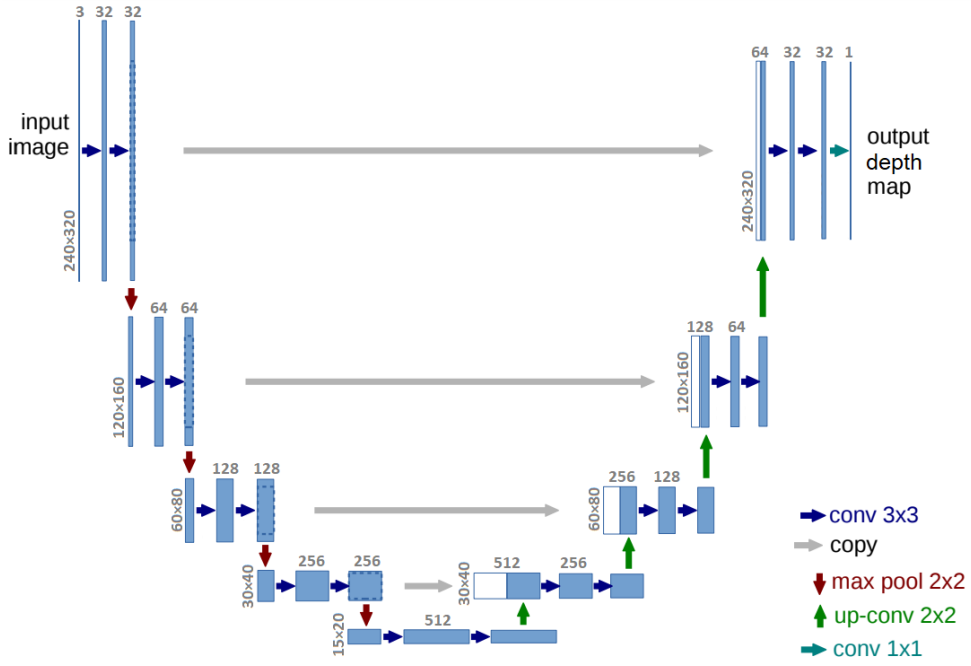
استاد درس: دکتر فاطمی‌زاده

بهمن ۱۴۰۰

تخمین عمق:

برای شبکه تخمین عمق از مدل استفاده شده در این پروژه استفاده می‌کنیم. این مدل مبتنی بر دو مقاله [۱] و [۲] طراحی شده‌است.

معماری شبکه: برای تخمین عمق، از یک شبکه انکدر-دیکدر U-net استفاده می‌شود. این شبکه از skip-connection استفاده می‌کند و در اصل به منظور segmentation طراحی شده‌است. انکدر از ۴ بلوک متوالی مشابه تشکیل شده‌است. هر بلوک از دو لایه کانولوشنی 3x3 با تابع فعالساز Leaky ReLU با پارامتر $\alpha = 0.2$ و به دنبال آن یک لایه 2x2 max-pooling ایجاد می‌شود. در هر بلوک تعداد کانال‌های ویژگی ۲ برابر کانال‌های بلوک قبلی انتخاب می‌شود. دیکدر نیز ۴ بلوک مشابه دارد. در هر بلوک، ابتدا ورودی بلوک قبلی از یک لایه up-sampling 2x2 با کانولوشن 3x3 عبور می‌کند و سپس با خروجی همتای خود در قسمت انکدر concatenate می‌شود. دو لایه کانولوشنی مشابه بلوک‌های انکدر نیز به دنبال آن می‌آید. بر خلاف انکدر تعداد feature map ها در هر بلوک دیکدر نصف بلوک قبلی است. پس از تمام لایه‌های کانولوشنی دیکدر و انکدر یک لایه batch-normalization قرار دارد. در نهایت نیز یک لایه کانولوشنی 1x1 با فعالساز sigmoid به منظور تولید خروجی با ابعاد دلخواه قرار می‌گیرد. لازم به ذکر است که رزولوشن تصاویر اصلی و عمق‌ها پیش از ورود به شبکه به نصف کاهش می‌یابد. شکل زیر معماری این شبکه را نمایش می‌دهد.



تابع هزینه: در این مدل تابع هزینه‌ای تعریف می‌شود که با کمینه کردن اختلاف عمق‌ها و همزمان جریمه کردن اعوجاج جزییات با فرکانس بالا در تصاویر عمق، نوعی تعادل میان این تصاویر تولیدی ایجاد کند. جزییات فرکانس بالای مذکور، نوعا مربوط به مرز اشیا در تصاویر هستند.

اگر y تصویر عمق اصلی و $\hat{y}$ تصویر عمق تخمینی شبکه باشد، تابع هزینه از جمع ۳ عبارت تشکیل می‌شود:

$$L(y, \hat{y}) = \lambda L_{depth}(y, \hat{y}) + L_{grad}(y, \hat{y}) + L_{SSIM}(y, \hat{y})$$

عبارت اول هرینه L1 درایه‌به‌درایه روی مقادیر عمق است:

$$L_{depth}(y, \hat{y}) = \frac{1}{n}\sum_{p}^{n} |y_p - \hat{y}_p|$$

عبارت دوم هزینه L1 روی گرادیان تصاویر عمق است:

$$L_{grad}(y, \hat{y}) = \frac{1}{n}\sum_{p}^{n} |g_x(y_p, \hat{y}_p)| + |g_y(y_p, \hat{y}_p)|$$

در جایی که g_x و g_y به ترتیب در مؤلفه‌های x و y، اختلاف گرادیان تصاویر y و $\hat{y}$ را محاسبه می‌کنند.

عبارت سوم نیز از معیار Structural Similarity (SSIM) که معمولا برای کارهای بازسازی تصاویر به کار می‌رود استفاده می‌کند. از آن جایی که این معیار همواره کوچکتر از ۱ است تعریف می‌شود:

$$L_{SSIM}(y, \hat{y}) = \frac{1 - SSIM(y, \hat{y})}{2}$$

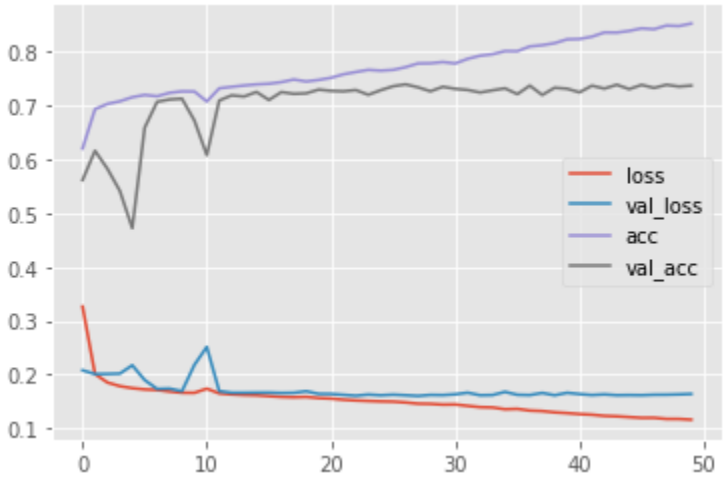
همچنین در نظر می‌گیریم: $\lambda = 0.1$.

Data Augmentation: به منظور جلوگیری از بیش‌برازش و عمومی شدن هرچه بیش‌تر شبکه از augmentation برای داده‌های آموزش استفاده می‌شود. چون شبکه، به منظور تخمین عمق یک تصویرِ کامل طراحی شده‌است، لزوما هر تبدیل هندسی مناسب نیست. چون ایجاد اعوجاج در دامنه تصاویر همیشه یک تفسیر معنادار برای عمق آن ندارد. اعمال یک گردش (flip) عمودی ممکن است کمکی به آموزش تصاویر داخل ساختمان نکند (برای مثال جابجایی سقف و کف یک اتاق را در نظر بگیرید). چرخش (rotation) نیز مشکل مشابهی دارد. بنابراین، تنها یک گردش افقی (آینه‌ای) روی تصاویر با احتمال ۵۰ درصد اعمال می‌شود.

معیار ارزیابی دقت: به این منظور ابتدا عمق واقعی هر تصویر و همچنین خروجی شبکه به صورت خطی بین ۰ و ۱ نرمالیزه می‌شود. سپس این مقادیر به ۰ یا ۱ گرد می‌شوند. آن گاه همانند یک شبکه طبقه‌بند، تعداد

تخمین‌های درست در هر تصویر به تعداد کل پیکسل‌های آن تقسیم می‌شود. عدد حاصل شده دقت شبکه برای آن تصویر است.

آموزش: برای آموزش از داده‌های برچسب‌دار (Labeled) مجموعه NYU Depth Dataset V2 استفاده می‌کنیم. این مجموعه شامل ۱۴۴۹ تصویر داخل ساختمان به همراه تصاویر عمق متناظر است. ۲۰ درصد داده‌ها به عنوان داده‌های ارزیابی و بقیه به عنوان داده‌های آموزش مورد استفاده قرار می‌گیرند. شبکه را ۵۰ epoch آموزش می‌دهیم. نتیجه آموزش به صورت زیر است:



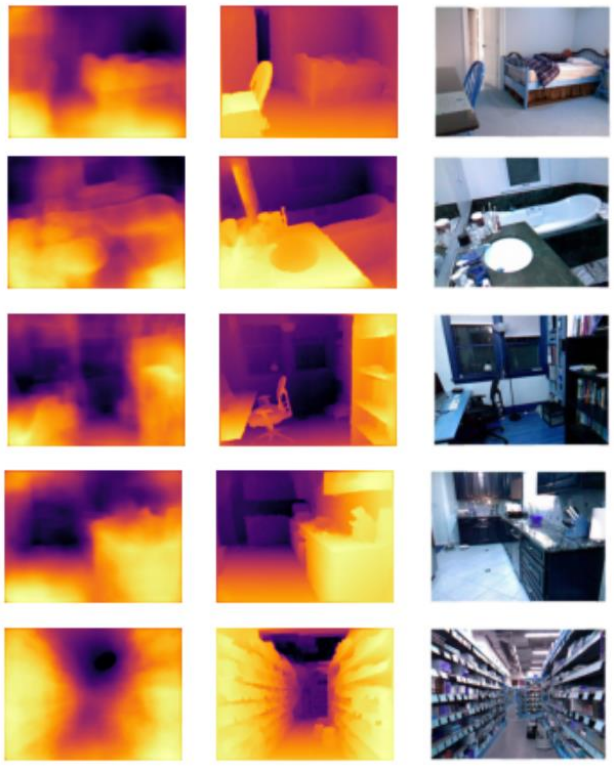
شکل ۱ نمودارهای آموزش شبکه تخمین عمق

شبکه به خوبی آموزش دیده و هیچ گونه overfitting دیده نمی‌شود. مقدار تابع هزینه برای داده‌های ارزیابی برابر ۰.۱۶۳۳ و مقدار دقت آن (با معیار معرفی‌شده) برابر ۷۳.۵٪ حاصل می‌شود.

```
loss: 0.1633 - depth_acc: 0.7351
```

ارزیابی کیفی:

۵ نمونه از تخمین شبکه روی داده‌های ارزیابی را مشاهده می‌کنیم.



شکل ۲: ارزیابی شبکه. تصاویر رنگی در ردیف راست، عمق واقعی در ردیف وسط و عمق تخمینی شبکه در ردیف چپ قرار دارند.

عمق‌ها به طور نسبی درست تشخیص داده شده‌اند و با توجه به داده‌های کم آموزش قابل قبول است!

دیگر شبکه‌های تخمین عمق موجود:

شبکه‌های دیگری نیز برای تخمین عمق با استفاده از تنها یک تصویر ارائه شده است. برای مثال در این لینک شبکه‌های مختلف دیگری مقایسه شده است. در مقایسه با شبکه استفاده شده در این پروژه، شبکه‌هایی که دقت بالاتری را گزارش کرده‌اند، پیچیدگی بیش از حد برای این کار را دارند و در این گام از شبکه معرفی شده استفاده شده است. بدیهی است در گام‌های بعدی می‌توان شبکه‌های دیگری را با شبکه استفاده شده جایگزین کرد. البته علاوه بر خطایی که برای این شبکه‌ها گزارش می‌شود، سرعت شبکه نیز معیار مهمی است زیرا علاقه‌مند هستیم در گام‌های آینده این شبکه را در کاربردهای real-time نیز استفاده کنیم.

منابع این بخش و مقالات بررسی شده برای استفاده در گام‌های بعد:

[1] U-Net: Convolutional Networks for Biomedical Image Segmentation

[2] High Quality Monocular Depth Estimation via Transfer Learning

Yu, J.; Choi, H. YOLO MDE: Object Detection with Monocular Depth Estimation. Electronics 2022, 11, 76.

Wang, H.-M.; Lin, H.-Y.; Chang, C.-C. Object Detection and Depth Estimation Approach Based on Deep Convolutional Neural Networks. Sensors 2021, 21, 4755

Deep Learning-Based Monocular Depth Estimation Methods—A State-of-the-Art Review .

مقالاتی که از دیتاست مورد نظر استفاده کرده‌اند:

https://paperswithcode.com/sota/depth-estimation-on-nyu-depth-v2

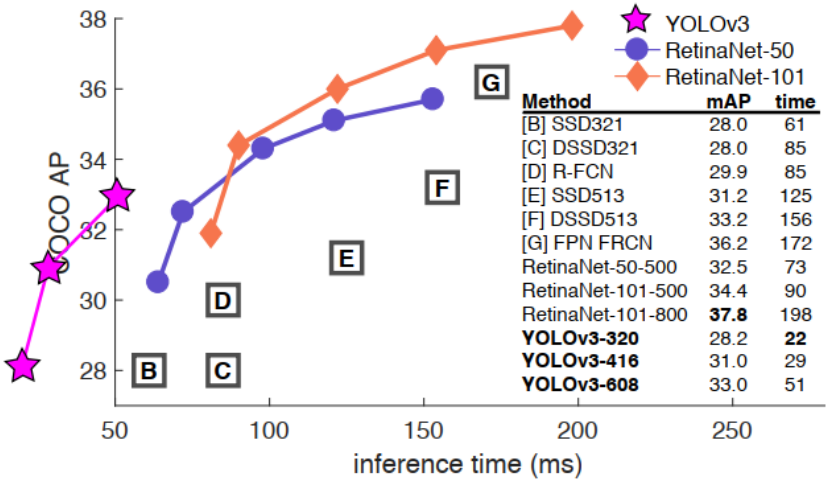
mind map برای شبکه‌های تشخیص عمق و مقالات مرتبط:

https://github.com/sxfduter/monocular-depth-estimation

از کاربردهای تشخیص عمق در سیستم‌های مربوط به خودروهای بدون سرنشین نیز می‌توان نام برد که با توجه به اهمیت موضوع، در مقالات مرتبط در آن حوزه نیز شبکه‌های مرتبط معرفی شده‌اند.

تشخیص اشیا (Object Detection):

در این قسمت، توضیحی در مورد قسمت تشخیص اشیا می دهیم. در این پروژه از شبکه از پیش آموزش داده شده YOLO v3 استفاده شده است. همان‌طور که مقاله مربوط به این شبکه گزارش کرده است این شبکه علاوه بر دقت بالا، سرعت بالایی نیز دارد که به ما کمک خواهد کرد در کاربردهای real-time پروژه از آن استفاده کنیم (*شکل ۳*). اگر چه ورژن‌های جدیدتر این شبکه نیز معرفی شده اند اما به علت کاربردهای مختلف این شبکه در حوزه‌های گوناگون و فراهم بودن پشتیبانی‌های بیشتر از آن، در این گام از این شبکه استفاده شده است. بدیهی است در گام‌های بعد می‌توان شبکه‌های دیگر تشخیص اشیا را نیز جایگزین کرد و مقایسه‌های مختلفی بین آن ها در این حوزه به خصوص انجام داد.

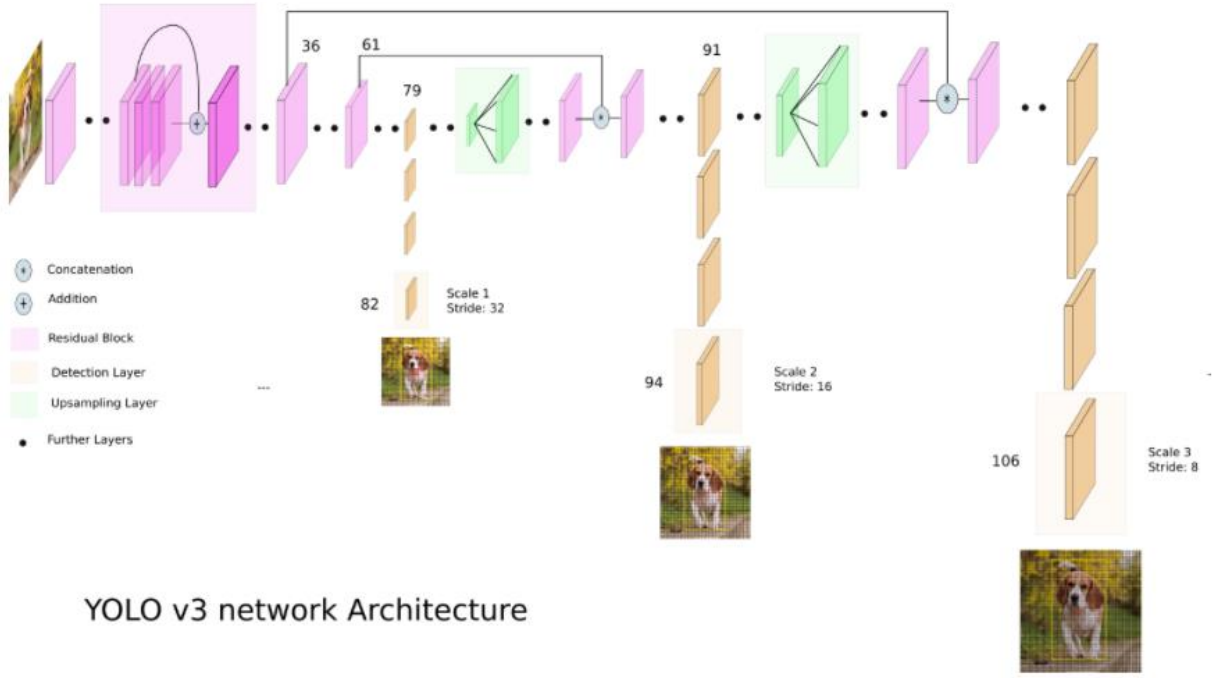


شکل ۳: مقایسه سرعت و دقت شبکه YOLOv3 با دیگر شبکه‌های تشخیص اشیا (Lin، ۲۰۱۷).

این شبکه مزایای متعددی در مقایسه با دیگر شبکه‌ها دارد. این شبکه به کل تصویر در زمان تست نگاه می‌کند و لذا پیش‌بینی‌ها بر مبنای محتوای کل تصویر خواهد بود. این شبکه در مقایسه با شبکه‌هایی مثل R-CNN تنها یک تصویر را برای خروجی نیاز دارد نه چندین تصویر که منجر به سرعت بالای آن در مقایسه با R-CNN و Fast R-CNN شده است.

شبکه قبلی YOLO (YOLO v2) در ساختار خود از شبکه عمیق darknet-19 استفاده می‌کند. شبکه‌ای ۱۹ لایه که با ۱۱ لایه دیگر برای تشخیص اشیا تجهیز شده است. نکته مغفول مانده در این شبکه residual block ها،

skip connection ها و up sampling بود. در ابتدا YOLO v3 با استفاده از darknet با ۵۳ لایه به همراه ۵۳ لایه دیگر استک شده بر آن، (۱۰۶ لایه) ارائه شد که در مقایسه با YOLO v2 آهسته تر بود.

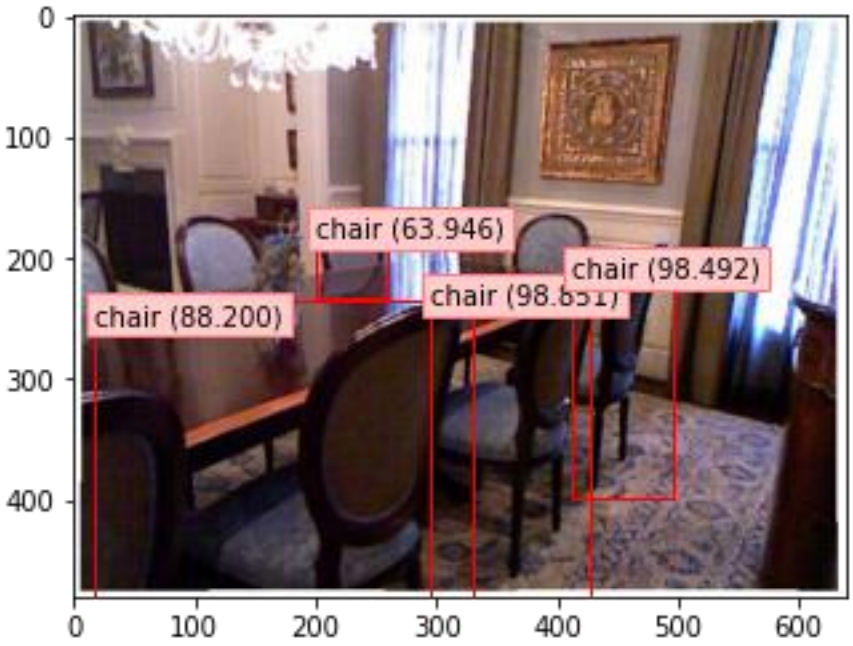


شکل ۴: ساختار اولیه YOLO v3 (۲).

تشخیص هر شی با استفاده از این شبکه، در ۳ scale متفاوت انجام می‌شود (down sample کردن ورودی با ۳۲، ۱۶ و ۸). این اتفاق این قابلیت را به شبکه می‌دهد که اشیا کوچک نیز خوب تشخیص داده شوند در مقایسه با YOLOv2. در این شبکه هم‌چنین دیگر از softmax در خروجی استفاده نشد تا بتوان کلاس‌های مختلف مثل person و woman را نیز طبقه‌بندی کرد. در این ورژن از لاجیستیک رگراسیون و یک ترشهولد استفاده می‌شود. کلاس‌های با امتیاز بالاتر از ترشهولد به هر box اختصاص داده می‌شود.

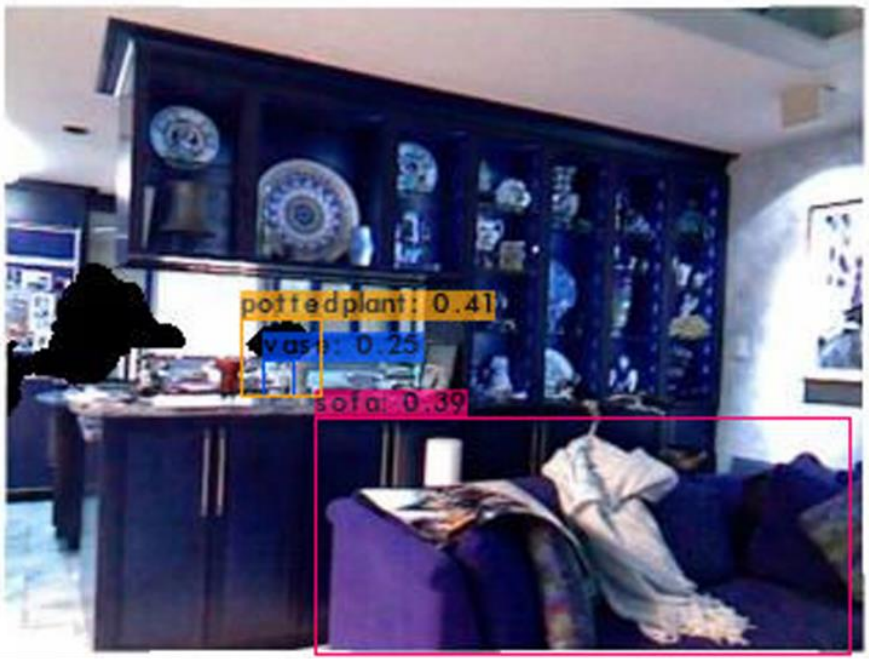
پیاده‌سازی: در این پروژه از لینک قرار داده شده برای پیاده‌سازی شبکه YOLOv3 در Tensorflow استفاده شده است. پس از تعریف لایه‌های مختلف شبکه darknet (که شبیه VGG-16 نیز هست)، وزن آموزش داده شده بر دیتاست coco برای هر قسمت لود شده است. با توجه به دیتاست خواسته شده در پروژه آموزش مجددی بر این شبکه انجام نشده است. پس از آن تصاویر دیتاست مورد نظر پس از تغییر سایز به سایز ورودی شبکه (۴۱۶ در ۴۱۶)، به شبکه داده می‌شوند و اشیا در آن تشخیص داده می‌شود.

برای مثال یک خروجی برای تصویر ۱۴۴۸ در شکل ۵ آورده شده است. در کنار هر box، لیبل مربوطه و درصد تعلق آن کلاس به آن box نیز نوشته شده است.



شکل ۵: خروجی نمونه شبکه تشخیص اشیا.

لازم به ذکر است تشخیص اشیا به کمک YOLO v4 نیز انجام شده است و در فایل YOLO_Object_Detection.ipynb قابل مشاهده است (یک خروجی از آن در شکل ۶ آوره شده است.). به خاطر پیچیدگی خروجی گرفتن از آن برای اتصال دو شبکه سراغ YOLO v3 رفتیم. در گام‌های بعدی می‌توان از این شبکه نیز در این قسمت استفاده کرد.



شکل ۶: خروجی شبکه YOLO v4 برای قسمت تشخیص اشیا.

مقالات و منابع این بخش:

YOLOv3: An Incremental Improvement, Joseph Redmon, Ali Farhadi, 2018.

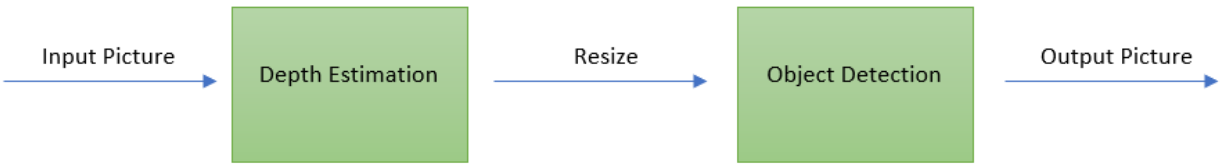
https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b

در خصوص شبکه YOLO v4:

https://medium.com/aiguys/yolo-v4-explained-in-full-detail-5200b77aa825

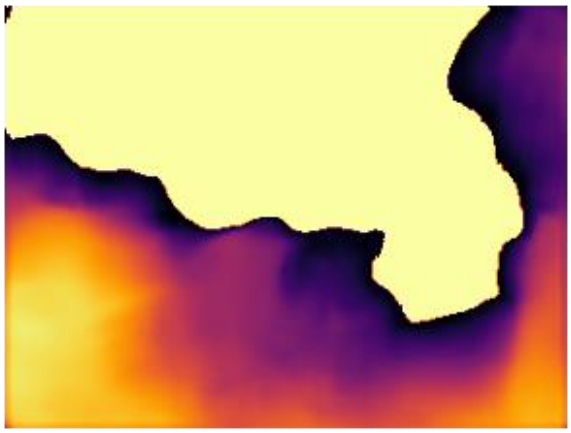
تشخیص عمق و تشخیص اشیا به صورت سری:

با توجه به دستور پروژه، دو شبکه ذکر شده برای تشخیص عمق و تشخیص اشیا در کنار یکدیگر قرار گرفته‌اند. می‌توان حالت‌های مختلفی از ترکیب این دو شبکه را ارائه کرد. در این پروژه شبکه تشخیص عمق ابتدا و شبکه تشخیص اشیا پس از آن مطابق *شکل ۷* به هم متصل شده‌اند. در واقع با استفاده از خروجی شبکه تخمین عمق، در عمق مورد نظر کاربر، اشیا تشخیص داده می‌شود. از مزایای این ساختار می‌توان به load کمتر برای شبکه تشخیص اشیا اشاره کرد. این دو شبکه را به صورت موازی نیز می‌توان به کار گرفت به طوری که خروجی تشخیص عمق و تشخیص Object در انتها تجمیع شده و برای هر box یک عمق نسبت داده شود و در انتها عمق مد نظر کاربر تنها در نظر گرفته شود که منجر به کمی افزایش دقت و لود بیشتر Object Detection خواهد شد. هم‌چنین چون دو شبکه را می‌توان به صورت موازی به کار گرفت در کاربرد real-time با سخت افزار مناسب این روش بهتر خواهد بود. به علت خواست پروژه مبنی بر اتصال سری شبکه‌ها این حالت مورد بررسی بیشتر قرار نگرفته است اما در گام‌های آینده برای استفاده از این شبکه در کاربرد real-time این ساختار پیشنهاد می‌شود. ساختار دیگر قابل استفاده آن است که شبکه تشخیص اشیا ابتدا و سپس شبکه تخمین عمق قرار بگیرد که به علت استفاده شبکه تخمین عمق از ویژگی‌های تمام تصویر این روش نیازمند ارتباطات بیشتری بین این دو شبکه و یا ورودی و شبکه تخمین عمق خواهد بود.



شکل ۷: اتصال دو شبکه به صورت سری.

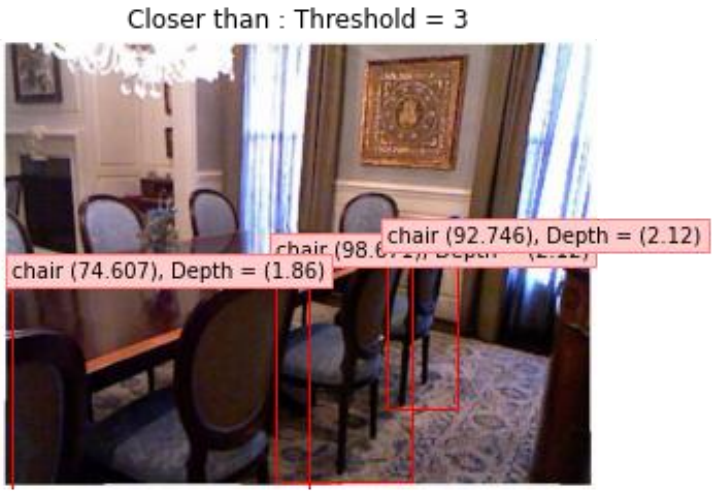
در ساختار استفاده شده، خروجی شبکه تخمین عمق با عمق مد نظر کاربر ماسک می‌شود (*شکل ۸*) و بر تصویر اصلی اعمال می‌شود (*شکل ۹*) و پس از تغییر اندازه به شبکه تشخیص اشیا داده شده و خروجی نهایی بدست می‌آید.



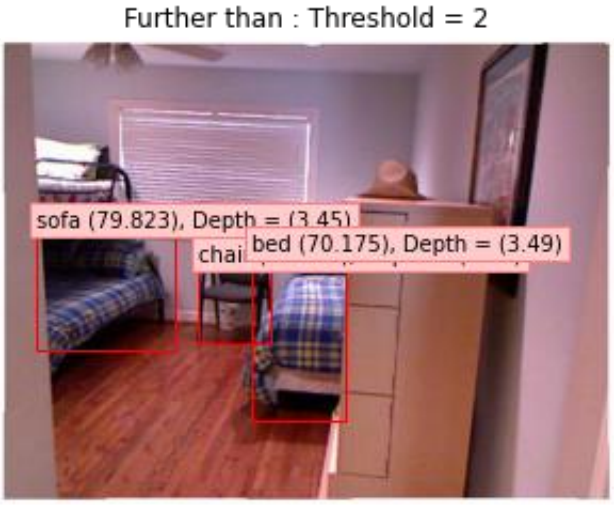
شکل ۸: خروجی شبکه تشخیص عمق به صورت ماسک شده با توجه به ورودی کاربر.

شکل ۹: تصویر ماسک شده به جهت ورود به شبکه تشخیص اشیا، عمق نزدیک تر از ۳ متر.

در نهایت خروجی شبکه تشخیص اشیا، خروجی مطلوب در این پروژه را به ما می‌دهد. لازم به ذکر است خروجی عمق برای هر box از میانگین عمق‌های مربوطه در خروجی شبکه تشخیص عمق بدست می‌آید. در *شکل ۱۰* و *شکل ۱۱* دو نمونه خروجی مربوط به دورتر از عمق دلخواه و نزدیک تر از آن برای دو تصویر متفاوت رسم شده است.



شکل ۱۰: خروجی نهایی برای نزدیک تر از ۳ متر.



شکل ۱۱: خروجی نهایی برای دورتر از ۲ متر.

گام‌های بعدی قابل اجرا: جایگزینی شبکه‌های تشخیص عمق و تشخیص اشیا با شبکه‌های با دقت و سرعت بالاتر (YOLO v4 و دیگر شبکه‌ها)، اعمال شبکه بر ویدئو، تست شبکه به صورت real-time، بررسی شبکه بر روی دیگر دیتاست‌ها.

ارزیابی شبکه joint:

در ارزیابی شبکه‌های joint، به صورت معمول با دست داشتن دیتاستی که مناسب عملکرد دو یا چند شبکه موجود در مجموعه شبکه مورد نظر باشد، می‌توان ارزیابی مربوطه را انجام داد. در این پروژه به طور خاص این امکان وجود ندارد زیرا task مربوط به object detection از داده‌های دیتاست قابل ارزیابی نیست و در مرحله آموزش نیز به وزن‌های از پیش آموزش داده شده با دیتاست دیگری بسنده شده است. در این قسمت مقالات مربوط به این حوزه و روش‌های ارزیابی مختلفی که می‌توان با آن شبکه‌های joint را ارزیابی کرد مورد بررسی قرار می‌گیرد.

کلیدواژه‌های مفید: Stacked Neural Networks، Joint Learning، Unifying and Merging trained Networks، Ensemble، Multi Modal، Multi Task.

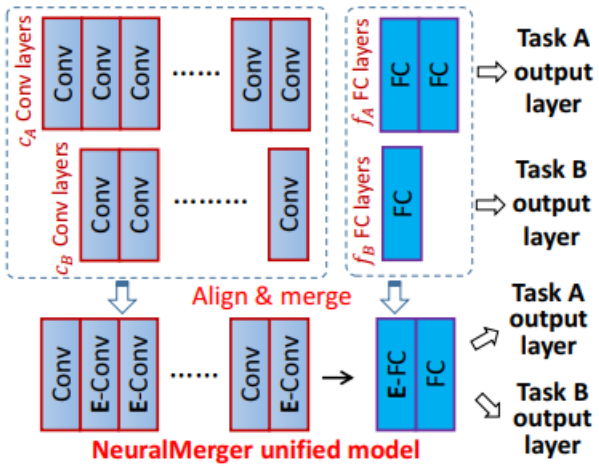
مقالات و کاربردهای مربوط به Ensemble با ذکر این عنوان که استفاده از شبکه‌های مختلف باعث می‌شود خروجی بهتری داشته باشیم و فیچرهای متنوع‌تری استخراج شوند، این راهکار را در بسیاری از موارد پیشنهاد داده‌اند. آموزش شبکه‌های Ensemble شده به صورت End-to-End راهکاری است که باعث می‌شود وزن‌های شبکه‌ها قویا correlated شود. مقاله (۱ و ۲) به این مبحث پرداخته است و استفاده از شبکه‌های از پیش آموزش داده شده به صورت مستقل را در کنار آموزش End-to-End بررسی می‌کند. در آموزش joint شبکه‌ها با استفاده از تابع هزینه مشابه زیر که در مقاله (۱) آورده شده است می‌توان آموزش شبکه‌ها را همزمان انجام داد:

$$L_\lambda \stackrel{\text{def}}{=} \lambda \mathrm{D}(p \,\|\, \bar{q}) + (1-\lambda)\frac{1}{M}\sum_{j=1}^{M} \mathrm{D}(p \,\|\, q_j)\,,$$

در این رابطه D، مقدار KL-divergence است. این دو مقاله نتیجه گرفته‌اند که برای مدل‌های ساده‌تر آموزش joint کمک بهتری می‌کند و در مدل‌های پیچیده‌تر بهتر است هر شبکه به صورت مجزا آموزش ببینند.

مقاله ۳ در تلاش برای unified کردن دو شبکه well-trained که دو تسک مختلف را بر عهده دارند، ساختاری را معرفی کرده است که در آن با share کردن وزن‌ها در مرحله inference، دو شبکه را به هم متصل می‌کند. این وزن‌ها را نیز می‌توان سپس fine-tune کرد. ساختار مربوطه در شکل ۱۲ آورده شده است. در این مقاله صوت و تصویر، پوشش و جنسیت مورد بررسی قرار گرفته است و در رده مقالات مربوط به multi modal و multi task

می‌توان آن را طبقه‌بندی کرد. در قسمت آموزش و تابع هزینه نیز با در نظر گرفتن وزنی برای هر کدام از ترم‌های مربوط به هر یک از مدل‌ها، تابع هزینه نوشته شده است.

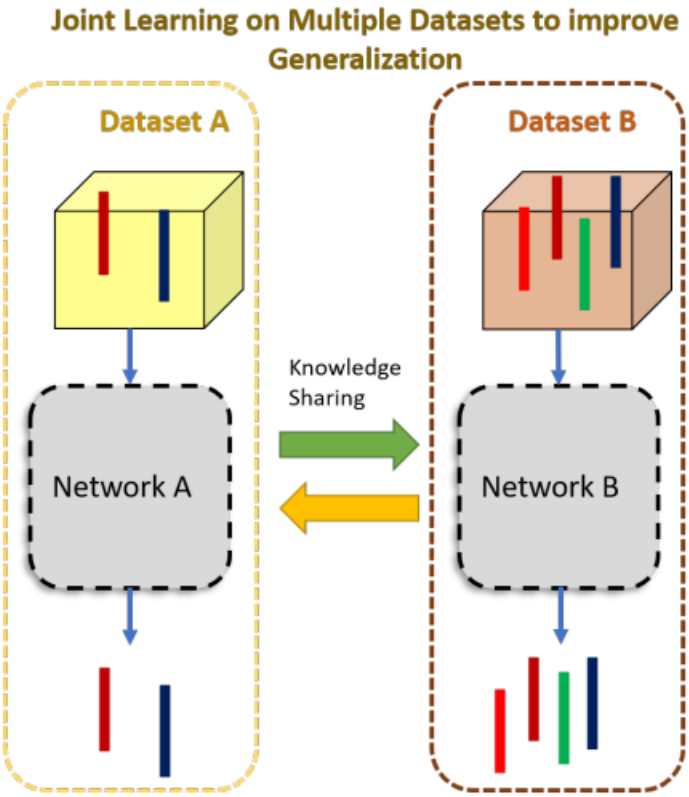


شکل ۱۲: ساختار ارائه شده در مقاله ۳.

در نمونه دیگر، مقاله ۴ با بررسی مبحث Multi Modal، دقت شبکه خود را در مقایسه با State of the Art مورد نظر در هر Modal، نزدیک گزارش کرده است. این در حالی است که وزن‌ها و هایپرپارامترهای مسئله در این مقاله tune نشده اند برای هر تسک به صورت مجزا. این نتیجه در این مورد می‌تواند این فرضیه را تقویت کند که خروجی شبکه Multi Modal تفاوت چندانی با شبکه تک Modality ندارد. در این پروژه نیز اگر این نتیجه را در نظر بگیریم، دقت تشخیص اشیا joint ، در مقایسه با دقت YOLO تفاوت چندانی نخواهد داشت که البته بهتر است بر روی دیتاستی که بتواند شبکه joint را ارزیابی کند، این موضوع بررسی شود.

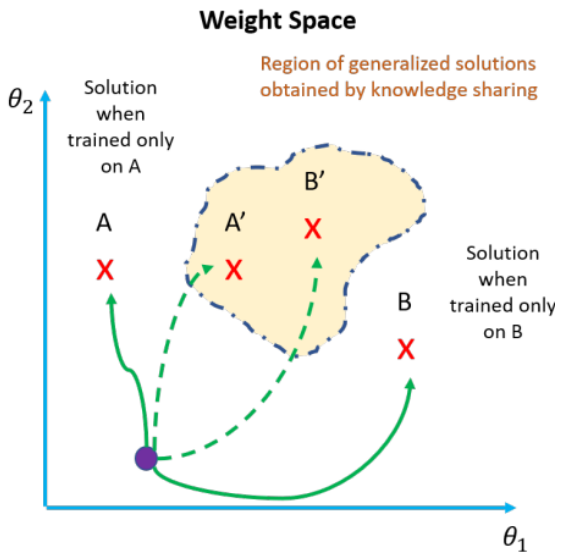
مقاله ۵ به بررسی Joint Learning دو شبکه که بر روی دو دیتاست متفاوت آموزش دیده‌اند پرداخته است. این کار به نحوی دیتاست بزرگتری را سعی کرده است در شبکه‌ها در نظر بگیرد. در این مقاله اشاره شده است که استفاده از دو دیتاست مجزا برای آموزش شبکه‌ها به صورت مجزا (همان‌کاری که در این پروژه انجام شده است) می‌تواند منجر به آن شود تغییر کوچکی در دیتاست باعث overfit شدن شبکه کل شود اگر چه بر روی شبکه خود تابع هزینه مقدار کمی را دارد. در این مقاله با اشاره به این موضوع، بین دو شبکه ارتباطاتی پیشنهاد شده است تا این دو شبکه بتوانند از هم یاد بگیرند. این کار برای افزایش تعمیم پذیری شبکه پیشنهاد شده است (شکل ۱۳). از این ایده می‌توان

در شبکه مربوط به پروژه نیز استفاده کرد به طوریکه بین لایه‌های شبکه تخمین عمق در بحث تشخیص اشیا و بالعکس ارتباط ایجاد کنیم.



شکل ۱۳: برقراری ارتباط بین دو شبکه مجزا با دیتاست مجزا برای افزایش تعمیم پذیری.

با اشتراک گذاشتن وزن‌ها بین دو شبکه فرض بر آن است که فضایی وجود دارد که وزن‌ها می‌توانند با تنظیم مناسب در آن فضا قرار بگیرند تا مدل joint تعمیم پذیری بیشتری داشته باشد (شکل ۱۴). در مسئله مربوط به ما نیز همین چالش‌ها برقرار است. با ساختار سری دو شبکه، ورودی شبکه دوم (خروجی شبکه اول) در فضایی که شبکه دوم در آن قرار داشته است در نظر گرفته می‌شود. خروجی شبکه اول هر چه بیشتر در این فضا قرار داشته باشد، دقت شبکه دوم قابل دسترس‌تر و معتبرتر است. لذا به نحوی می‌بایست نگاشت خروجی شبکه اول را در فضای شبکه دوم مورد بررسی قرار داد.

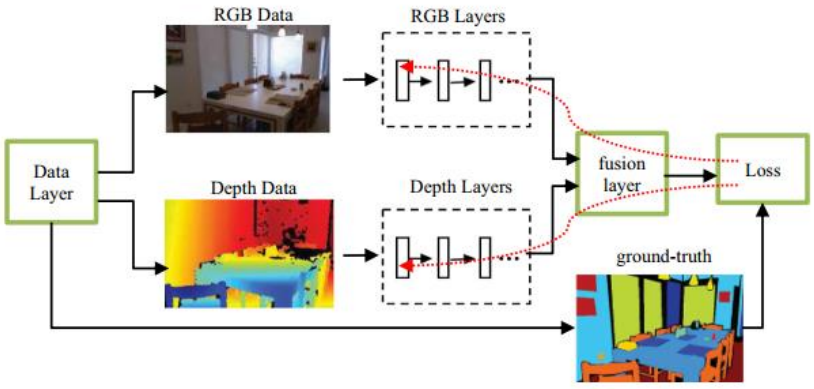


شکل ۱۴: فضای وزن‌ها، فرض بر آن است که با اشتراک گذاشتن وزن‌ها در قسمتی از این فضا می‌توان به تعمیم‌پذیری بهتری دست یافت.

در ادامه این مقاله با ارائه دادن تابع هزینه، آموزش شبکه‌ها مورد بررسی قرار گرفته است.

مقاله ۶ نیز به بحث stack کردن شبکه‌ها پرداخته است. در این بحث، شبکه‌های مختلف با یک ورودی آموزش می‌بینند و در انتها امتیاز مربوط به هر شبکه محاسبه شده و Ensemble می‌شود.

مقاله ۷ که بر روی دیتاست این پروژه کار کرده است، با استفاده از تصویر RGB و تخمین عمق، Segmentation را انجام داده است. این کار با توجه به آن که ground truth در اختیار است قابل انجام است در صورتی که object detection بر روی این دیتاست به این شیوه امکان پذیر نیست.



شکل ۱۵: ساختار ارائه شده در مقاله ۵.

منابع و لینک‌های مفید این قسمت:

1 - To Ensemble or Not Ensemble: When does End-To-End Training Fail? Andrew M. Webb, 2020

2- Joint Training of Neural Network Ensembles, Andrew M. Webb, University of Manchester.

3- Unifying and Merging Well-trained Deep Neural Networks for Inference Stage, Yi-Min Chou, 2018

4- One Model To Learn Them All, Lukasz Kaiser, 2017.

5- Joint Learning for Spatial Context-based Seismic Inversion of Multiple Datasets for Improved Generalizability and Robustness, Ahmad Mustafa, 2021

6- S-NN: Stacked Neural Networks, Milad Mohammadi, Stanford University, 2016.

7- RGB-D joint modelling with scene geometric information for indoor semantic segmentation, Hong Liu, 2018

8- https://en.wikipedia.org/wiki/Multi-task_learning