انجمن علمی دانشکده مهندسی کامپیوتر | نشریه علمی **بایت** | شماره ۱

بایت

RANDOMIZED SMOOTHING
A Brief Introduction to Certified ML

LINUS TROVALDS

CE GROUPS
An Intro to Computer Engineering Scientific Groups

OPEN-SOURCE PROJECTS
How to Contribute

UDP HOLE PUNCHING

OSINT
Open-Source Intelligence

XAI
Explainable AI

لینوس تروالدز

Lovely Bits Get Together ...

Securrty Torrtoraed

بایت

سلام!

به نشریه‌ی بایت خوش اومدین.

این متنی که الان دارید می‌خونید، حاصل ماه‌ها تلاش تیم نشریه‌ست تا بتونیم بایت رو بعنوان اولین نشریه و جامعه‌ی علمی دانشکده ثبت کنیم. داستان بایت و ایده‌ی این جامعه‌ی علمی (community) دانشجویی، از یکی از درس‌های دانشکده شروع شد. درسی که قطعا همه‌ی ما باهاش خاطره داریم: مبانی برنامه‌نویسی! در ترم پاییز ۹۹ درس مبانی، یک فایل جزوه‌مانند از مطالب مهم درس منتشر می‌شد به اسم «کدنامه» که از اون استقبال زیادی شد؛ چون خیلی مختصر و مفید مطالب درس رو توضیح می‌داد. بعد از اون، هدف کدنامه فراتر رفت.

دومین ترمی که کدنامه نوشته می‌شد، برای درس «برنامه‌سازی پیشرفته» بود که یک تیم مخصوص از تدریس‌یارهای درس، برای آماده‌کردن محتوای اون تلاش می‌کردن. این‌بار، علاوه بر مطالب مرتبط به درس، مطالب فراتر از درس و راجع به توانایی‌های دیباگ، مهندسی نرم‌افزار، کاربرد مفاهیم مختلف درس در کارهای صنعتی و ... هم به کدنامه اضافه شد. در ترم‌های بعد از اون، کم‌کم ساختار کدنامه به کلی تغییر کرد و تبدیل شد به جایی برای تبادل و ارائه دانش: در قالب متن، درمورد مطالب فراتر از درس و نکات جذابی که هرکس خودش اون‌هارو کشف کرده بود و می‌خواست با بقیه در میون بذاره. با طی این فرآیند، هر روز بیشتر و بیشتر جای خالی یک جامعه‌ی علمی و فضایی برای تبادل آزاد دانش درمورد حوزه‌های مختلف علوم کامپیوتر در دانشکده‌مون حس می‌شد. درنهایت، از طریق انجمن علمی دانشکده، پیگیر شدیم که چنین جای خالی‌ای رو پر کنیم! مسیر سختی بود اما از حق نگذریم، همه می‌دونیم که همیشه، «اولین» بودن سخته! با گذر از همه‌ی اتفاقاتی که خیلی فشرده و خلاصه براتون تعریف کردیم، اولین شماره‌ی نشریه‌ی بایت به دستتون رسیده.

ما تازه اول این راه هستیم و کلی اتفاقات جذاب و هیجان‌انگیز پیش‌روی ماست! اگر شما هم یکی از علاقمندان به انتشار دانش و بحث‌های علمی هستین و دوست دارین چیزی که اخیرا یاد گرفتین رو با بقیه هم درمیون بذارین، شما عضو بایت هستین! خوش اومدین. امیدواریم از بایت لذت ببرین.

Linus Torvalds

عواملی که باعث می‌شوند شما کارهایی را انجام دهید که یک موجود زنده می‌کند:

اولی بقاست، دومی نظم اجتماعی و سومی تفریح.

ایمان محمدی و فاطمه حریرفروش

لینوس بندیکت تورووالدز (Linus Benedict Torvalds)، با همین جمله کتاب زندگی‌نامه‌اش را آغاز می‌کند. لینوس، یکی از بزرگ‌ترین شخصیت‌های دنیای فناوری اطلاعات و نرم‌افزارهای آزاد و متن‌باز، در تاریخ ۲۸ دسامبر ۱۹۶۹ در هلسینکی (پایتخت فنلاند)، متولد شد. او در خانواده‌ای با علاقه و توجه ویژه به حوزه ژورنالیسم و رسانه‌ها پرورش یافت. در حالی که دیگر کودکان شهر هلسینکی وقت خود را با اسکی یا هاکی همراه با خانواده یا دوستان خود می‌گذراندند، لینوس با کنجکاوی زیاد در دنیای کامپیوترها، تمام تلاش خود را به کار گرفت تا بفهمد آن‌ها چگونه عمل می‌کنند. به عنوان مثال، با استفاده از کدهای مورس (Morse code) و تبدیل آنها به زبان ماشین (Machine Language)، خودش را سرگرم می‌کرد و زندگیش کاملاً مشغول به صفر و یک‌های دنیای کامپیوتر بود. در این دوران، لینوس حدود ۱۲ یا شاید ۱۳-۱۴ سال داشت و همچنان به کاوش های جدید در دنیای کامپیوتر ادامه می‌داد.

مرد ریاضی

لینوس دوره دبیرستان خود را در مدرسه نورسن گذراند. این مدرسه بین پنج دبیرستان سوئدی زبان در شهر هلسینکی واقع بود و در آن زمان به عنوان مرکزیت آموزشی شناخته می‌شد. علاقه زیادی به درس‌های ریاضی و فیزیک داشت. او به قدری علاقه‌مند به این دروس بود که در مدرسه به عنوان «مرد ریاضی» شناخته می‌شد. لینوس در سال ۱۹۸۸، تحصیلات دانشگاهی خود را در دانشگاه هلسینکی (University of Helsinki) آغاز کرد. این دانشگاه در آن زمان یکی از برجسته‌ترین دانشگاه‌های فنلاند در زمینه علوم کامپیوتر بود. از قول خودش، اولین سال از دوره هفت‌ساله لیسانس برای او بسیار پر حاصل بود. کتاب‌ها و قطعات کامپیوتری به عنوان عناصر کلیدی تحصیلی و خلاقیت او به کار می‌رفتند. از مطالعه کتاب‌های فیزیک گرفته تا توسعه برنامه‌های بازی جدید و حتی سفر به دنیای سه‌گانه علمی-تخیلی کتاب‌های داگلاس آدامز (Douglas Adams)، او به خودآموزی و تجربه گسترده در عرصه علوم کامپیوتر و فیزیک مشغول بود.

خیلی از ما تاریخ را با شغل‌ها، مکان‌های زندگی، و یا خاطراتی که ساخته‌ایم به یاد می‌آوریم. برای لینوس، سال‌های زندگی او با کامپیوترها به یاد ماندنی است. در دوران نوجوانی او، هنوز کامپیوترهای شخصی به میزان گسترده‌تری وجود نداشتند. اما لینوس یک کامپیوتر خانگی که از پدربزرگش به ارث برده بود داشت. این کامپیوتر، مشابه دیگر کامپیوترهای آن دوران، با استفاده از زبان اسمبلی (Assembly language) برنامه‌نویسی می‌شد. لینوس هزینه‌ی کامپیوتر دوم خود را با پس‌انداز پول از کریسمس، هدیه تولد و همچنین به عنوان تمیزکار در پارک هلسینکی فنلاند کار کرد و از جوایز مختلف مدرسه از جمله لقب «مرد ریاضی»، توانست تهیه کند.

سرانجام، کامپیوتر IBM که یک ماه برای بازی استفاده شده بود، به آخرین ورژن وسیله‌ی جذاب زندگی برای لینوس تبدیل شد! برای درک عمیق‌تر شخصیت لینوس، معتقدیم که نگاهی به زندگی او از این زاویه کمک می‌کند:

> *یا یک پروژه‌ای برای انجام داریم و وقت خود را صرف آن می‌کنیم، یا اگر پروژه‌ای موجود نیست، انگیزه‌ای برای برنامه‌نویسی و کار با کامپیوتر وجود ندارد.*

او وقت خود را به جز برنامه‌نویسی، در فعالیت‌های دیگری همچون بازی‌های بولینگ و اسنوکر با دوستان، بازی با گربه‌ها و گشت‌زدن در

خانه با حوله‌ی کهنه‌اش سپری می‌کند. حتی دلیل اصلی رفتن به خدمت سربازی نیز همین علاقه به برنامه‌نویسی بود. هنگامی که دوره سربازی ۱۱ ماهه او به پایان رسید و وی وارد دانشگاه شد، انتظارش برای شروع کلاس‌های برنامه‌نویسی زبان C و سیستم‌عامل یونیکس (Unix) فرا رسید، علاقه‌ی لینوس به سیستم‌عامل یونیکس بیشتر شد و با مفاهیم آن آشنا شد. او به فلسفه‌ی نهفته در سیستم‌عامل یونیکس علاقه‌مند شد و روزانه وقت خود را در کنار این مفاهیم گذراند. او به عبارتی معتقد بود:

> *برای حل مسائل پیچیده، نیاز به روابط پیچیده نیست. کافی است آجرهایی داشته باشیم که با آن‌ها هر چیزی بسازیم، مانند زبان انگلیسی که با آن چیزهای پیچیده‌ای را بیان می‌کنیم و نه زبان چینی که پیچیدگی‌های زیادی دارد.*

این فلسفه او، انگیزه‌اش برای شروع پروژه‌های جدید را ایجاد کرد.

تولد لینوکس

در آن دوران، علاوه بر یونیکس، سیستم‌عامل‌های دیگری نیز در بازار موجود بودند، از جمله سیستم‌عامل‌های شبه یونیکس مانند مینیکس (Minix) که بسیار کارآمد بودند. لینوس از نسخه اولیه مینیکس بر روی جدیدترین کامپیوتر خود نصب کرد. اما در آن زمان، هیچ کس نمی‌پنداشت که این شبیه‌ساز ترمینال مینیکس می‌تواند جوانه‌ای برای یک پروژه بزرگتر باشد. لینوس با شبیه‌ساز ترمینال مینیکس مشکلاتی را تجربه کرد و تصمیم گرفت که آن را به زبان ماشین بازنویسی کند. این پروژه بسیار پیچیده بود، زیرا نیازمند برنامه‌نویسی در پایین‌ترین سطح کامپیوتر بود. این باعث شد تا لینوس دقیقاً به نحوه عملکرد پردازنده مرکزی آگاه شود. او برای یادگیری درباره CPU، این برنامه را با استفاده از زبان اسمبلی نوشت و به قول خودش، نوشتن یک برنامه به زبان اسمبلی در مقایسه با نوشتن همان برنامه با زبان C، نیاز به ۳۰ برابر زمان بیشتر داشت!

تغییر ذهنیت لینوس از یک شبیه‌ساز ترمینال به سمت یک سیستم‌عامل رخ داد زمانی که نیاز به ارسال و دریافت فایل‌ها و ذخیره‌ی آن‌ها را حس کرد و به درایور دیسک نیاز داشت که در کنار شبیه‌ساز ترمینالش نبود. در بهار سال ۱۹۹۱، در یک برنامه کاری فشرده، لینوس شروع به اضافه کردن قابلیت‌های جدید به پروژه خود کرد که به طور ناخودآگاه منجر به ایجاد یک سیستم‌عامل جدید شد. سال‌ها بعد از تولید این سیستم‌عامل جدید، پس از تعدادی از پیشرفت‌های آن، لینوس تصمیم گرفت که این دست‌آوردها را با جامعه به اشتراک بگذارد. او اطلاعات خود را با گروه خبری مینیکس به اشتراک گذاشت و با افرادی (هر چند محدود) روبرو شد که تمایل داشتند به آزمون نسخه بتای سیستم عامل لینوکس (Linux) بپردازند. این اتفاق مارک‌های شروع اپن‌سورس شدن سیستم‌عامل لینوکس را قرار داد و جامعه بزرگی از توسعه‌دهندگان و کاربران را به خود جلب کرد. شاید اگر هر کدام از این آجرهای مسیر زندگی لینوس، طور دیگری قرار می‌گرفتند، شبیه‌ساز ترمینالی که برای ارتباط سریع‌تر و راحت‌تر با سایت دانشگاه طراحی کرده بود، حالا به یک سیستم‌عامل متن‌باز و محور اصلی دنیای خیلی از برنامه‌نویس‌ها تبدیل نمی‌شد.

امیدواریم این متن کوتاه در انبوه متن‌های مربوط به لینوس و لینوکس، مقدمه‌ی خوبی برای خواندن کتاب «فقط برای تفریح: داستان یک انقلاب اتفاقی» به قلم خود لینوس و ترجمه‌ی جادی میرمیرانی باشد. لینوس در متن کتاب به این موضوع اشاره دارد که کتاب طراحی و پیاده‌سازی سیستم‌های عامل (نوشته تننبام، متحول‌کننده‌ی زندگی او بود. شاید شما هم از کتاب لینوس با این عنوان یاد کنید.

معرفی گروه‌های علمی دانشکده

۳۸ سال قبل، دانشکدهٔ مهندسی کامپیوتر دانشگاه شریف کار خود را با راه‌اندازی رشته‌های کارشناسی «نرم‌افزار» و «سخت‌افزار» شروع کرد. ۲ سال بعد، ورودی‌های کارشناسی ارشد در گرایش‌های «معماری کامپیوتر» و «مهندسی نرم‌افزار» به دانشکده اضافه شدند. از آن زمان تابه‌حال، دانشکده به اقتضای زمان، دستخوش اضافه‌شدن گرایش‌ها و گروه‌های آموزشی متعددی شده است.

در حال حاضر در مقطع کارشناسی، با ادغام «نرم‌افزار» و «سخت‌افزار»، رشتهٔ «مهندسی کامپیوتر» ارائه می‌شود. همچنین ۸ گرایش در مقطع ارشد و ۶ گرایش در مقطع دکترا، در قالب ۴ گروه «معماری کامپیوتر»، «مهندسی نرم‌افزار»، «هوش مصنوعی» و «فناوری اطلاعات»، ارائه می‌شود. در ادامهٔ متن به معرفی کوتاهی از این گروه‌ها می‌پردازیم. در شماره‌های بعدی بایت به جزئیات فعالیت‌های علمی اساتید و آزمایشگاه‌ها خواهیم پرداخت.

عماد امام‌جمعه و امیرحسین حسن‌زاده

نرم‌افزار

مهندسی نرم‌افزار شاخه‌ای از مهندسی کامپیوتر است که در مورد طراحی، توسعه، تست (آزمون نرم‌افزار)، نگهداری و مدیریت نرم‌افزارهای کامپیوتری صحبت می‌کند.

وظیفهٔ من به عنوان یک مهندس نرم‌افزار چیست؟

شما مسئولیت تمام مراحل تولید نرم‌افزار، از تحلیل نیازمندی‌ها[۱] تا طراحی ، پیاده‌سازی[۲]، تست و نگهداری را به عهده دارید! در واقع شما فقط یک برنامه‌نویس نیستید! شما به عنوان یک مهندس نرم‌افزار، باید برنامه‌ای بنویسید که خوانایی بالا و قابلیت نگهداری داشته باشد. (تفاوت شما با یک برنامه‌نویس معمولی در اینجاست.)

استادها و آزمایشگاه‌های گروه نرم‌افزار چطوری‌اند؟

اینجا توضیح اجمالی‌ای از زمینهٔ تحقیقاتی چند تا از آزمایشگاه‌های دانشکده را برای‌تان نوشته‌ایم که اگر به زمینهٔ خاصی علاقه دارید، بتوانید بر اساسش تصمیم‌گیری کنید. البته که فعالیت‌های آزمایشگاه‌ها به چیزهایی که نوشته‌ایم محدود نمی‌شوند و این توضیحات را بر اساس علایق پژوهشی[۳] اساتید و پرس‌وجو از دانشجوها جمع‌آوری کرده‌ایم:

اگر به حوزهٔ تست نرم‌افزار علاقه دارید، آزمایشگاه دکتر میریان در این زمینه فعالیت می‌کند؛ اگر به حوزهٔ داده‌کاوی[۴]، گراف و یا نظریهٔ بازی‌ها علاقه‌مند هستید، اساتیدی مثل دکتر فضلی و دکتر ایزدی در این زمینه‌ها فعالیت دارند؛ اگر حوزه‌های تئوری را بیشتر می‌پسندید، دکتر موقر در این زمینه آزمایشگاه خوبی دارند؛ دکتر رامسین، در حوزه‌های متدولوژی[۵] و مهندسی نرم‌افزار فعالیت دارند؛ در نهایت، دانشجوهای دکتر حبیبی هم در حوزه‌های مالی و مقداری هم یادگیری ماشین و هوش مصنوعی فعالیت می‌کنند.

معماری

معماری کامپیوتر شاخه‌ای از مهندسی کامپیوتر است که به طراحی سیستم‌های کامپیوتری برای اهداف مختلف و بهینه‌سازی عملکرد[۶] می‌پردازد. به‌کارگیری بهینهٔ سخت‌افزار و استفاده از تکنیک‌های هوشمندانهٔ نرم‌افزاری و الگوریتمی، بخش عمدهٔ این شاخه را شامل می‌شود. البته گروه معماری کامپیوتر در دانشکدهٔ ما، کمی تعریف گسترده‌تری از معماری کامپیوتر عادی دارد و شاخه‌های دیگری را هم شامل می‌شود.

۱ Requirements Analysis
۲ Implementation
۳ Research Interests
٤ Data Mining
٥ Data Mining
٦ Performance Optimization

چه موضوعات داغی در این شاخه وجود دارند؟

کامپیوترهای کوانتومی، سخت‌افزارهای ویژه[۷]، پردازنده‌های گرافیکی[۸] و تنسوری[۹]، مهندسی نورومورفیک[۱۰] و رایانش نوری[۱۱] از حیطه‌های فعال و پرجنب‌وجوش تحقیقاتی این شاخه‌اند.

در دانشکده چه خبر است؟

اینجا هم مثل گروه‌های دیگر، توضیح اجمالی‌ای از حیطهٔ تحقیقاتی چندتا از آزمایشگاه‌های دانشکده در این گروه را آورده‌ایم:

دکتر ایزدی بیشتر در حوزهٔ سیستم‌های توزیع‌شده[۱۲] کار می‌کنند؛ دکتر حسابی و دکتر اسدی بیشتر در حوزهٔ ذخیره‌سازی[۱۳] اطلاعات فعالیت می‌کنند؛ دکتر سربازی روی سیستم‌های چند هسته‌ای[۱٤] کار می‌کنند؛ دکتر کوهی در زمینه‌های پردازش نوری و کمی هم کامپیوترهای کوانتومی تحقیق می‌کنند؛ در زمینهٔ سیستم‌های نهفته[۱٥] هم دکتر انصاری فعالیت می‌کنند.

فناوری اطلاعات

فناوری اطلاعات[۱٦] به استفادهٔ کامپیوتر برای ساخت، تحلیل، ذخیره‌سازی، استخراج و تبادل هرگونه اطلاعات گفته می‌شود. IT در واقع زیرمجموعه‌ای از ICT یا همان Information and Communication Technology است. شبکه و امنیت از مهم‌ترین مباحث این شاخه تحقیقاتی هستند.

از فناوری اطلاعات در دانشکدهٔ خودمان چه خبر؟

دکتر صفایی در زمینهٔ اینترنت اشیا[۱۷] کار می‌کنند؛ دکتر امینی و دکتر جلیلی در حوزه‌های مختلف امنیت و شبکه فعالیت می‌کنند؛ دکتر بیات در زمینهٔ رمزنگاری و 5G به تحقیق مشغول‌اند؛ دکتر خرازی هم در حیطهٔ سیستم‌عامل فعالیت می‌کنند.

لازم به ذکر است که گروه فناوری اطلاعات با ۴ استاد، کمترین تعداد استاد در هیئت علمی دانشکده را دارد.

۷ Specialized Hardwares
۸ Graphics Processing Unit (GPU)
۹ Tensor Processing Unit (TPU)
۱۰ Neuromorphic Engineering
۱۱ Optical Computing
۱۲ Distributed Systems
۱۳ Storage
۱٤ Multi-core Systems
۱٥ Embedded Systems
۱٦ Information Technology
۱۷ Internet of Things (IOT)

هوش مصنوعی

احتمالاً تابه‌حال شنیده‌اید که هوش مصنوعی[۱] چیست؛ اما تعریف دقیق هوش مصنوعی به این شکل است: هوش مصنوعی توانایی ماشین‌ها به تقلید یا تقویت خرد انسانی گفته می‌شود؛ مثل نتیجه‌گیری یا یادگیری از تجربه. سیستم‌های هوش مصنوعی اطراف ما روی کار، رفتار، مصرف رسانه، امنیت و ... تأثیر بسزایی دارند. شاید بتوان گفت شناخته‌شده‌ترین زیر شاخهٔ هوش مصنوعی در دنیای امروز، یادگیری ماشین[۲] است.

از هوش مصنوعی کجا استفاده می‌شود؟

استفاده از هوش مصنوعی در صنعت و علم در سال‌های اخیر رشد چشمگیری داشته‌است. از معروف‌ترین کاربردهای هوش مصنوعی می‌توان به موتورهای جست‌وجو[۳]، سیستم‌های توصیه‌کننده[۴]، فهم زبان‌های طبیعی[۵]، ماشین‌های خودران و هوش مصنوعی مولد[۶] اشاره کرد.

حوزه‌های رایج[۷] هوش مصنوعی چه شکلی‌اند؟

سرعت رشد و پیشرفت هوش مصنوعی بسیار زیاد بوده و زمینه‌های مطرح همواره در حال عوض شدن‌اند. اما از مطرح‌ترین زمینه‌ها می‌توان به موارد زیر اشاره کرد:

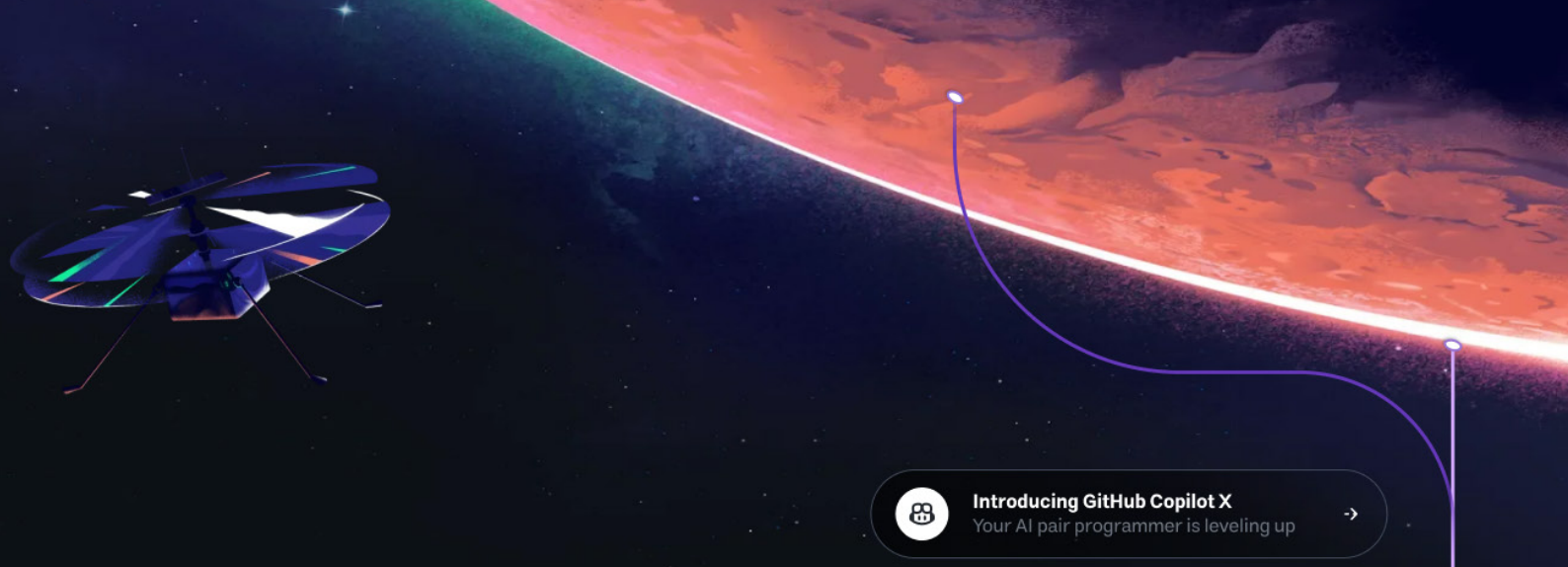
- مدل‌های زبانی بزرگ[۸]
- یادگیری عمیق[۹]
- بینایی رایانه‌ای[۱۰]
- پردازش زبان طبیعی

وضعیت هوش مصنوعی در دانشکده چگونه است؟

دکتر شریفی زارچی به همراه دکتر مطهری در حوزهٔ بیوانفورماتیک[۱۱] مشغول به کارند؛ دکتر کسایی بیشتر در زمینهٔ پردازش تصویر فعالیت می‌کنند؛ دکتر سلیمانی در زمینهٔ یادگیری عمیق فعال هستند؛ در حیطهٔ پردازش زبان طبیعی هم دکتر بیگی و دکتر عسگری مشغول به فعالیت‌اند؛ دکتر ربیعی هم در حوزهٔ شبکه‌های اجتماعی[۱۲] فعالیت می‌کنند. دکتر رهبان هم در حوزه یادگیری عمیق و همچنین تصاویر پزشکی فعالیت می کنند.

---

۱ Artificial Intelligence (AI)
۲ Machine Learning (ML)
۳ Search Engines
٤ Recommendation System
٥ Natural Language Understanding (NLU)
٦ Generative AI
۷ Trends
۸ Large Language Models
۹ Deep Learning
۱۰ Computer Vision
۱۱ Bioinformatics
۱۲ Social Network



چگونه به پروژه‌های متن‌باز کمک کنیم؟

امیرمهدی نامجو

آیا تابه‌حال به کلمهٔ متن‌باز یا Open Source برخورد کرده‌اید؟ اصطلاح Open Source به شکل ساده، به برنامه‌هایی گفته می‌شود که کد آن‌ها به صورت عمومی در اینترنت قرار دارد و افراد مختلف از همهٔ نقاط دنیا می‌توانند به توسعه و بهبود آن کمک کنند. حتی اگر به‌طور مستقیم با ابزارهای متن‌باز در پروژه‌های خود کار نکرده باشید، قطعاً به‌طور غیرمستقیم در زندگی شما نقش داشته‌اند. به‌عنوان مثال بخش عمدهٔ مرورگرهای امروزی بر پایهٔ Chromium هستند که توسط گوگل به عنوان یک پروژه متن‌باز ایجاد شده و افراد زیادی از سراسر دنیا به مقدار کم یا زیاد در توسعهٔ آن نقش داشته‌اند.

چرا متن‌باز؟

شاید برای‌تان سوال باشد که اصلاً چرا یک گروه باید پروژهٔ خود را به صورت متن‌باز ایجاد کند، درحالی‌که اگر کد آن را تنها برای خود نگه دارد، احتمالاً به دلیل انحصار می‌تواند سود بیش‌تری ببرد؟ دلیل این موضوع این است که هرچند پروژه‌های متن‌بسته به دلیل انحصار می‌توانند در زمینه‌ای، سودآوری بیش‌تری داشته باشند، اما پروژه‌های متن‌باز نیز می‌توانند مزایای چشمگیری برای سازندگان آن به ارمغان آورند. یکی از مهم‌ترین مسائل، این است که افراد زیادی کد شما را می‌بینند و می‌توانند مشکلات امنیتی و دیگر مشکلات آن را برطرف کنند. این موضوع می‌تواند باعث ایجاد جامعه‌ای از علاقه‌مندان به آن محصول شود که ممکن است در آینده، بدنهٔ اصلی توسعه‌دهندگان آن محصول را تشکیل دهند و حتی به شرکت تولیدکنندهٔ آن بپیوندند. عموماً به دلیل همین شفافیت، اعتماد به پروژه‌های متن‌باز بسیار بیش‌تر است. به علاوه، توسعه‌دهندگان کوچک و همچنین برنامه‌نویسان می‌توانند از طریق به اشتراک گذاشتن پروژه‌های خود به صورت متن‌باز، بدون نیاز به تشکیل یک شرکت بزرگ، از مزایای کمک‌های دیگران بهره‌مند شوند.

از سوی دیگر، سوالی که مطرح می‌شود این است که کمک به پروژه‌های متن‌باز چه سودی برای افراد دارد، یا به بیان بهتر، من چرا باید به این پروژه‌ها کمک کنم؟ برای این سوال جواب‌های متعددی وجود دارد. اولین مورد این است که با کمک به برنامه‌های متن‌باز، عملاً می‌توانید به بهبود برنامه‌هایی که خودتان از آن‌ها استفاده می‌کنید کمک کنید. علاوه بر این، مهارت‌های شما در زمینه‌های فنی نیز تقویت می‌شود و می‌توانید دانش خود را گسترش دهید. همچنین، در خلال این کار و در حین بحث با دیگر افرادی که به پروژه‌های یکسانی کمک می‌کنند، می‌توانید با افرادی با علائق مشترک آشنا شده و در پروژه‌های دیگر هم با آن‌ها همکاری داشته باشید. در کنار این‌ها کمک به پروژه‌های متن‌باز و ساخت این نوع پروژه‌ها می‌تواند به‌مرورزمان باعث شناخته‌تر شدن شما در این جوامع و عملاً ساخت یک رزومهٔ عملی از مهارت‌های شما بشود که می‌تواند در فرایند استخدام در شرکت‌های بزرگ، به‌شدت کمک‌کننده باشد؛ علی‌الخصوص که بسیاری از شرکت‌های مشهور نیز پروژه‌های بزرگی را به صورت متن‌باز مدیریت می‌کنند.

چگونه مشارکت کنیم؟

حال این سوال مطرح می‌شود که به چه شکل‌هایی می‌توان به پروژه‌های متن‌باز کمک کرد؟ با توجه به اینکه شمایی که این متن را می‌خوانید، احتمالاً در حوزهٔ علوم یا مهندسی کامپیوتر تحصیل یا فعالیت می‌کنید، اولین موردی که به ذهن می‌رسد این است که مشکلات کدهایی که وجود دارند و گزارش شده‌اند را برطرف کنید یا اقدام به پیاده‌سازی قابلیت‌های درخواستی برای یک پروژهٔ متن‌باز بکنید. این کمک به‌صورت برنامه‌نویسی، می‌تواند در حد برطرف کردن باگ‌های جزئی که خودتان یا دیگران کشف کرده‌اید تا کارهای بزرگی نظیر بازنویسی بخش‌های مهمی از پروژه، باشد. بااین‌حال تنها راه کمک به پروژه‌های متن‌باز، کدنویسی نیست و راه‌های متعدد دیگری برای این مسئله وجود دارد.

یکی از راه‌های دیگر برای کمک به پروژه‌های متن‌باز، بررسی مستندات، تکمیل، اصلاح و ویراستاری آن‌ها است. سازندگان اصلی پروژه، عموماً وقت بیش‌تری را روی خود کد می‌گذرانند اما بخش‌های جانبی، نظیر مستندات در بسیاری از مواقع نادیده گرفته می‌شود؛ درحالی‌که مستندات بخش بسیار مهمی هستند که در طول زمان حتی اهمیتی بیش‌تر از خود کد پیدا می‌کنند.
ازطرف‌دیگر، می‌توان به بررسی خود برنامه‌ها و رفع ایرادات جزئی در برنامه، نظیر غلط‌های نگارشی و املایی و یا به مشکلاتی که در ترجمه از زبان‌های مختلف ایجاد می‌شود، پرداخت. علاوه‌بر آن، می‌توان به افرادی که در بخش Issues پروژه‌ها سوال پرسیده‌اند کمک کرد. حتی در ابعاد گسترده‌تر گاهی شاهد برگزاری رویدادهایی توسط طرفداران جامعهٔ متن‌باز در جهت گسترش این فرهنگ و کمک به آن بوده‌ایم.

علاوه‌بر این‌ها، نیاز نیست که حتماً به پروژه‌هایی که از پیش وجود دارند کمک کنید. می‌توانید خودتان پروژه‌های مفید یا جدیدی را شروع کنید و یا پروژه‌های قبلی خود را به صورت متن‌باز قرار دهید. حتی نیازی نیست که پروژهٔ شما لزوماً یک پروژهٔ نرم‌افزاری یا فنی باشد. بسیاری از پروژه‌های متن‌باز، راهنماها و یا لیست‌های تجمیعیِ مفید از مطالب مختلف هستند که می‌توانند در یک موضوع خاص کمک‌کننده باشد و همهٔ این‌ها می‌توانند در دستهٔ کمک به جامعهٔ متن‌باز در نظر گرفته شوند.

اینا قبول، ولی

چطور باید شروع کنم؟

۱

برای شروع کمک به پروژه‌های متن‌باز، بهترین کار استفاده از سایت معروف گیت‌هاب (Github) است. سایت گیت‌هاب، معروف‌ترین سایت برای پروژه‌های متن‌باز است و براساس سیستم مدیریت‌پروژهٔ گیت کار می‌کند؛ که خود پروژه‌ای متن‌باز است. یادگرفتن گیت به شکل مقدماتی برای کمک به جامعهٔ متن‌باز ضروری است ولی خوشبختانه منابع زیادی برای یادگرفتن آن وجود دارد و یادگرفتن مفاهیم آن به شکل ابتدایی بسیار ساده است.

۲

بعد از ساخت یک حساب کاربری در سایت گیت‌هاب، می‌توانید هم پروژه‌های خود را به صورت عمومی قرار دهید و هم به پروژه‌های دیگران کمک کنید. مشکلات و درخواست‌هایی که برای پروژه‌های مختلف وجود دارند، عموماً تحت‌عنوان Issue در گیت‌هاب قرار می‌گیرند. برای پیدا کردن Issueهای مناسب برای شروع، می‌توان از قابلیت‌های جست‌وجوی گیت‌هاب به خوبی بهره برد. گیت‌هاب هم امکان جست‌وجو براساس زبان‌های برنامه‌نویسیِ مورد استفاده در پروژه را برای کاربران مهیا می‌کند، هم اینکه می‌توان با برچسب‌هایی که روی Issueها قرار می‌گیرد، Issueهای مناسب را پیدا کرد. معمولاً Issueهایی که برای شروع مناسب هستند، برچسب «good first issue» را دارند. همین‌طور برچسب‌هایی نظیر Documentation برای Issueهایی که نیاز به کد زدن ندارند، استفاده می‌شوند. علاوه‌بر این، یک راه خوب برای پیدا کردن Issueهای مناسب، اضافه کردن contribute/ به انتهای آدرس پروژه‌های مختلف در گیت‌هاب است. با این کار صفحه‌ای شامل لیستی از Issueهایی که برای شروع مناسب هستند، نمایش داده می‌شود.

۳

در نهایت لازم است برای انتخاب پروژه‌ها به چندین مورد دقت کنید. اولین مورد بررسی فایل‌های README، LICENSE و CONTRIBUTING و در صورت وجود CODE_OF_CONDUCT در پروژه است تا متوجه بشوید که پروژه از چه لایسنسی استفاده می‌کند و چه قوانینی برای کمک به آن وجود دارد. علاوه‌بر آن با نگاهی به Issueها می‌توان فهمید که آیا صاحبان پروژه با سرعت خوبی پاسخگو هستند یا پروژه، یک پروژهٔ فراموش‌شده است. همچنین می‌توان به فضای گفت‌وگوی بین افراد مختلف در پروژه دقت کرد تا متوجه شوید که آیا جامعهٔ آن پروژه، فضایی دوستانه دارد یا نه.

در این مقاله سعی کردیم به‌طور خلاصه، پروژه‌های متن‌باز را معرفی کرده و به مزایای کمک به این پروژه‌ها و روش‌هایی برای کمک به آن‌ها بپردازیم. در نهایت مهم‌ترین نکته‌ای که باید به آن توجه کنید، این است که نترسید. برای کمک کردن به پروژه‌های مختلف لازم نیست بر تمامی بخش‌های آن تسلط داشته باشید. در پروژه‌های بزرگ، افراد انگشت‌شماری هستند که بر بخش‌های زیادی از پروژه تسلط دارند. جذابیت و زیبایی دنیای متن‌باز تا حد زیادی در این است که کاربران می‌توانند به صورت جزئی به پیشرفت یک پروژه کمک کنند؛ این کمک‌ها قطره قطره جمع می‌شوند و در نهایت منجر به پیشرفت‌های بزرگ در پروژه می‌شوند.

لینک‌های مفید

ریپوهایی برای شروع
- بدون برنامه‌نویسی
- بدون نیاز به برنامه‌نویسی

چگونگی کمک به جامعهٔ متن‌باز
- لینک اول
- لینک دوم

انواع لایسنس
- لینک اول

شروعی برای تازه‌کارها
- لینک اول
- لینک دوم
- لینک سوم

گیت‌هاب

گیت

UDP HOLE PUNCHING

هیربد بهنام

در دنیای امروز، زندگی ما به اینترنت گره خورده است و تقریبا تمامی کارهایمان را با اینترنت انجام می‌دهیم: حرف زدن با دوستان، خرید کردن از فروشگاه‌ها، بازی‌های کامپیوتری و ... یکی از چالش‌های افرادی که در حوزه‌ی شبکه‌های کامپیوتری کار می‌کنند، کم کردن ترافیک و به طبع کمتر شدن توان پردازشی مورد نیاز است که به کمک آن بتوانند با هزینه‌ی کمتر، به افراد بیشتری سرویس بدهند.

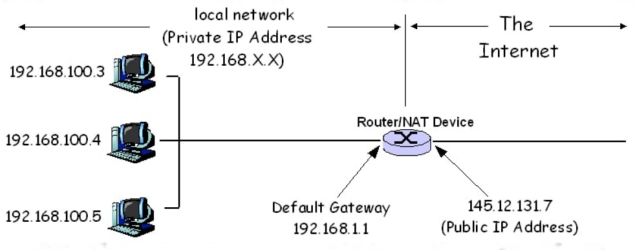
فرض کنید که یک سرویس تماس تصویری و صوتی آنلاین مثل گوگل‌میت یا اسکای‌روم دارید. اولین روشی که برای برقراری ارتباط بین دو فرد به ذهن می‌رسد این است که ترافیک مورد نظر به کمک یک سرور به هر فرد فرستاده شود یا به اصطلاح relay یا proxy شود. این کار مسلماً توان مصرفی زیادی را درگیر می‌کند و زمانی که تعداد کاربرهای ما بالا می‌رود، باید سرورهای قوی‌تری برای پراکسی کردن داده‌ها قرار دهیم. اما چه می‌شد اگر می‌توانستیم بدون یک سرور میانی دو شخصی که می‌خواهند با هم حرف بزنند را به هم وصل کنیم؟

امروزه بسیاری از سیستم‌عامل‌ها به یک فایروال مجهز هستند. این فایروال‌ها به صورت پیش‌فرض اجازه نمی‌دهند که هیچ کانکشنی از خارج شبکه شما به کامپیوتر شما زده شود. از طرفی دیگر، بسیاری از شبکه‌های خانگی پشت network address translation (NAT) هستند. NATها این اجازه را به کامپیوترهای متصل به آن می‌دهند که همگی آن‌ها یک IP مشترک داشته باشند. به عنوان مثال در صورتی که شما با کامپیوتر و گوشی خود که به مودم خودتان وصل هستند در گوگل عبارت «what's my ip» را جست و جو کنید IPهای یکسانی می‌گیرید.

چرا NAT؟

حال یک قدم عقب‌تر برویم و بررسی کنیم که چرا NAT؟ در دنیای امروزی اینترنت هر شخصی که می‌خواهد با دیگران ارتباط برقرار کند باید یک IP address داشته باشد. احتمالا آن‌ها را دیده‌اید: چهار عدد هستند که با ۳ نقطه از هم جدا شده‌اند. به عنوان مثال 81.31.168.91 IP سرور آموزش دانشگاه است. همچنین برای اینکه در خیلی از سیستم‌ها همزمان چندین اپلیکیشن بتوانند با یک IP ارتباط برقرار کنند یک عدد دیگر به اسم پورت برای هر IP تعریف می‌شود که نشانگر یک برنامه است که می‌تواند درخواست‌های از بیرون را قبول کند اینکه می‌خواهد با بیرون ارتباط برقرار کند. به عنوان مثال، 1.2.3.4:9200 می‌شود برنامه‌ای که بر روی پورت 9200 آی‌پی آدرس 1.2.3.4 است. استاندارد IPv4 تنها حدود ۲ میلیارد IP موجود دارد و در حال حاضر بیشتر از ۲ میلیارد وسیله متصل به اینترنت وجود دارد؛ یعنی به وضوح با کمبود تعداد آدرس مواجهیم! برای همین روشی به اسم NAT معرفی شد که به کمک آن می‌توان به چندین کامپیوتر یک IP را اختصاص داد.

این روش بدین صورت عمل می‌کند که شما باید به یک router (مسیریاب) یک IP بدهید که در اینترنت بتواند با بقیه حرف بزند. این همان IPای است که زمانی که در اینترنت سرچ می‌کنید what's my ip به آن بر می‌خورید. به این آدرس IP،آدرس Public IP می‌گویند. همچنین به هر یک از کامپیوترهایی که به شبکه‌ی داخلی شما وصل هستند یک آدرس Private IP می‌دهد. این IPها به نحوی یک IP مجازی هستند که در دنیای اینترنت به دستگاه خاصی داده نشده‌اند و همیشه برای استفاده‌ی NATها رزرو شده‌اند. به عنوان مثال IPهایی که با 192.168 یا 10 شروع می‌شوند از این دسته IPها هستند. زمانی که یکی از دستگاه‌های متصل به NAT درخواستی برای یک IP خارج از NAT (مثلا سایت گوگل) می‌خواهد بفرستد، دستگاه مجهز به NAT پکت‌هایی که در حال رفت و آمد از سمت گوگل به سمت کامپیوتر و برعکس است را بازنویسی می‌کند، طوری که آدرس فرستنده Public IP باشد. برای دیدن یک شمای کلی از NAT شکل زیر را نگاه کنید.



اما همچنان یک مشکل وجود دارد. فرض کنید که دو کامپیوتر که به NAT وصل هستند همزمان از یک پورت بخواهند که به یک سایت واحد متصل شوند. در اینجا نمی‌توان تنها IPها را تغییر داد چرا که درخواست دریافتی از سمت کامپیوتر مقصد معلوم نیست که برای کدام یک از کامپیوترهای local network هستند. در اینجا مجبور هستیم که آدرس port را نیز عوض کنیم.

به صورت کلی بسیاری از NATها دقیقا مثل پاراگراف بالا کار می‌کنند. فرض کنید که در ابتدا کامپیوتری با Private IP 10.1.1.5 می‌خواهد که از پورت 12345 یک پکت به پورت 54321 آی‌پی 2.3.4.5 بفرستد. زمانی که این پکت را سیستم‌عامل می‌سازد آدرس مبدا را همان 10.1.1.5 قرار می‌دهد. زمانی که NAT این بسته را دریافت می‌کند 10.1.1.5 و پورت 12345 را به آدرس Public IP و یک پورت رندوم تغییر می‌دهد و سپس بسته به اینترنت فرستاده می‌شود. همزمان NAT در ممووری خودش ذخیره می‌کند که در صورتی که یک پکت از 2.3.4.5:54321 آمد آن را برای 10.1.1.5:12345 بفرستد.

نکته‌ای که در اینجا وجود دارد این است که تنها بسته‌های دریافتی از 2.3.4.5:54321 به کامپیوتر مذکور منتقل می‌شوند. پس امکان ندارد که هر کسی در اینترنت بتواند برای کامپیوترهایی که پشت NAT هستند سرخود پکت بفرستد؛ حتما باید در ابتدا کامپیوتری که پشت NAT است با کامپیوتری که می‌خواهد ارتباط را برقرار کند اول یک پکت فرستاده باشد که NAT بداند که پکت‌های ورودی را به کجا ارسال کند. برای همین برقراری ارتباط مستقیم بین دو کامپیوتر که پشت NAT هستند امکان پذیر نیست...

یا شایدم بشه یه کاری کرد؟

فرض کنید که شما یک پکت به 1.2.3.4:1000 می‌فرستید. در این حالت NAT به خاطر می‌سپارد که هر چیزی پکتی از 1.2.3.4:1000 باید به کامپیوتر شما فرستاده شود. حال فرض کنید که دقیقا از همان IP و پورت کامپیوتر خودتان یک پکت به 5.6.7.8:2000 می‌فرستیم. حال NAT، همان IP + Port قبلی خودتان را به 5.6.7.8:2000 مپ می‌کند. پس اگر یک جوری دو کامپیوتری که پشت NAT هستند بتوانند برای IP و Port همدیگر یک پیام بفرستند، NATهایشان طوری تنظیم می‌شود که می‌توانند با هم به صورت مستقیم و بدون واسطه حرف بزنند!

به کمک همین موضوع ایده‌ی UDP Hole Punching مطرح شد. دلیل وجود اسم پروتکل UDP در این نام این است که این روش به خاطر stream base بودن TCP بر روی TCP جواب نمی‌دهد. (روش‌هایی برای اعمال این روش بر روی TCP نیز وجود دارد ولی باشد برای یک شماره دیگر!) در این پروتکل یک سرور دیگر نیز وجود دارد که باید از سمت هر دو کلاینت که می‌خواهد به هم وصل شوند قابل دسترسی باشد. اسم این سرور را STUN می‌نامیم. UDP Hole Punching بدین صورت عمل می‌کند که در ابتدا هر دو peer به STUN می‌گویند که می‌خواهیم با هم ارتباط برقرار کنیم. با این کار NAT هر دو peer به صورتی تنظیم می‌شود که پکت‌ها فقط بتوانند که به سرور STUN برسند و دریافت شوند. سپس STUN به هر کدام از peerها آدرس آن یکی peer را می‌دهد و هر یک از peerها به آن آدرس یک پکت می‌فرستند. با این کار NAT هر کدام طوری تنظیم می‌شود که می‌توانند برای همدیگر پیام بفرستند و در نتیجه دیگر سرور STUN به دردی نمی‌خورد؛ چرا که حال هر دو peer بدون واسطه می‌توانند با هم حرف بزنند.

خیری بدون شر؟ ...

اما هیچ خیری بدون شر نمی‌آید! UDP Hole Punching دو مشکل بزرگ دارد. اول از همه اینکه در بعضی نوع از NATها، به اسم Symmetric Nat، روش UDP Hole Punching کار نمی‌کند. در این نوع NATها با عوض شدن آدرس مقصد پورتی که NAT به عنوان پورت مبدا rewrite می‌کند نیز عوض می‌شود و برای همین آدرسی که دست STUN است با آدرسی که به آن یکی peer پیام فرستاده می‌شود فرق دارد. این نوع NAT معمولا در اپراتورهای تلفن همراه دیده می‌شود. یکی دیگر از مشکلات بزرگ UDP Hole Punching لو رفتن IP هر یک از peerها است. فرض کنید که شما می‌خواهید که IP آدرس یک شخص را پیدا کنید. برای این کار یکی از کارهایی که می‌توانید بکنید این است که به کمک یک برنامه‌ی مکالمه‌ی آنلاین مثل تلگرام* به آن شخص زنگ بزنید و با این کار UDP Hole Punching اتفاق می‌افتد و IP شخص مورد نظر بدست می‌آید.

* البته این موضوع را نیز در نظر بگیرید که تلگرام قابلیت خاموش کردن Peer-to-Peer voice call را دارد و به کمک آن می‌توانید از IP خود محافظت کنید.

OSINT

OPEN SOURCE INTELLIGENCE

کارآگاهی دروب!

معین آعلی

مقدمه‌ای بر OSINT

اوسینت(OSINT) یا Open Source IntTelligence به معنای جستجوی پیشرفته در منابع اطلاعاتی آزاد است. این مفهوم به مجموعه‌ای از روش‌ها، تکنیک‌ها و ابزارهایی اشاره دارد که به ما در جمع‌آوری، تحلیل و بررسی اطلاعات کمک می‌کند.

منابع اطلاعاتی‌ای در علم اوسینت استفاده می‌شوند که از منابع عمومی و علنی قابل دسترسی هستند. این منابع معمولاً شامل مواردی از قبیل وب‌سایت‌ها، شبکه‌های اجتماعی، گزارش‌های حکومتی، مقالات علمی، داده‌های عمومی، نقشه‌های جغرافیایی و... می‌شوند.

فرض کنید از شما خواسته می‌شود راجع به مواردی از قبیل «تعداد دانشجوهای ایرانی ساکن کانادا»، «شماره تلفن یک فرد خاص» یا «مکان‌هایی که می‌توان در آن‌ها یک سلبریتی خاص را ملاقات کرد» و مواردی مشابه. برای برخی از این سوال‌ها مانند «تعداد دانشجوهای ایرانی ساکن کانادا»، سرچ کردن داخل گوگل کافی است، اما برای پیدا کردن «مکان‌هایی که می‌توان در آن‌ها یک سلبریتی خاص را ملاقات کرد» سرچ گوگل کافی نیست! فرض کنید یک تصویر به شما می‌دهند و از شما می‌پرسند که این تصویر مربوط به کدام کشور، کدام شهر و کدام خیابان است؟ حال علم اوسینت معنا پیدا می‌کند و ما باید با استفاده از جست‌وجو در منابع اطلاعاتی خود به دنبال جوابی برای این پرسش‌ها باشیم.

تعریف دقیق OSINT

OSINT، به جستجو و تحلیل اطلاعات عمومی و آزاد از منابع عمومی در دنیای دیجیتال می‌پردازد. این نوع اطلاعات از منابعی مانند اینترنت، رسانه‌ها، شبکه‌های اجتماعی و منابع دیگر جمع‌آوری می‌شوند و به عنوان اطلاعات غیرمحرمانه و غیررسمی شناخته می‌شوند. OSINT به

تحلیل و تفسیر این اطلاعات به منظور بدست آوردن دیدی جامع‌تر در مورد یک موضوع می‌پردازد.
به زبان دیگر، OSINT می‌تواند مانند یک کارآگاه باشد که به دنبال کشف پاسخ به یک سوال از منابع عمومی می‌گردد. این نوع تحقیق به هر کسی که توانایی استفاده از تکنیک‌های جستجو را داشته باشد، اجازه می‌دهد که به سادگی و با هزینه معقول به اطلاعاتی دسترسی یابد که برای دیگران ارزشمند و گرانبها باشد. در OSINT، تنها به جستجوی اطلاعات عمومی پرداخته می‌شود و فعالیت‌های نفوذی و نقض حریم خصوصی افراد ممنوع می‌باشد.

کاربردها

استفاده‌های OSINT (استخراج اطلاعات متن‌باز) در امور مختلف بسیار گسترده است و در مختلف زمینه‌ها کاربرد دارد. یکی از کاربردهای مهم OSINT در دامنه امنیت سایبری است. سازمان‌ها و تیم‌های امنیتی از OSINT برای مانیتورینگ فعالیت‌های آنلاین، تشخیص تهدیدات سایبری، و اطلاع از آسیب‌پذیری‌های امنیتی در منابع متن‌باز بهره می‌برند. این اطلاعات می‌توانند به آن‌ها کمک کنند تا در مقابل حملات سایبری و نفوذهای مخرب مقاومت کنند.
همچنین، OSINT در حوزه جاسوسی صنعتی نقش مهمی ایفا می‌کند. شرکت‌ها و سازمان‌ها می‌توانند از این تکنیک برای جمع‌آوری اطلاعات درباره رقبا، بازارهای جدید، و فرصت‌های تجاری استفاده کنند. این اطلاعات به آن‌ها کمک می‌کند تا تصمیمات بهتری در زمینه استراتژی‌های کسب و کار بگیرند.
در ضمن، OSINT در زمینه رصد رفتار اجتماعی و جمعیت‌شناسی نیز کاربرد دارد. از این تکنیک می‌توان در مطالعات اجتماعی، تحلیل رفتار انسانی در شبکه‌های اجتماعی، و تشخیص الگوهای اجتماعی بهره‌برد. به طور کلی، OSINT به عنوان یک ابزار قدرتمند در تصمیم‌گیری‌ها و تحلیل‌های متعدد در زمینه‌های مختلف مورد استفاده قرار می‌گیرد.

چالش‌ها و مسائل اخلاقی

- استفاده از OSINT در مطالعات و فعالیت‌های تحقیقاتی با خود مسائل اخلاقی مهمی به همراه دارد. یکی از اصلی‌ترین مسائل اخلاقی مرتبط با OSINT حفظ حریم شخصی و حریم خصوصی افراد است. در تلاش برای جمع‌آوری اطلاعات از منابع عمومی، ممکن است افراد و اطلاعات حساس آن‌ها بدون اجازه به طور غیرمجاز مورد نظر قرار گیرند. این امر به نقض حقوق انسانی و حریم خصوصی افراد منجر می‌شود.
- مسئله دیگری که در مورد OSINT به وجود می‌آید، تشویش و تلاش برای تحت فشار قرار دادن افراد یا سازمان‌ها است. با انتشار اطلاعات دقیق یا نادقیق از منابع متن‌باز، ممکن است افراد یا سازمان‌ها به طور نادرست به عنوان مقصر در اتفاقات مشخصی معرفی شوند. این مسائل می‌توانند به نقض حقوق دیگران و آسیب به شخصیت و حریم خصوصی آن‌ها منجر شوند.

به طور کلی، اطلاعات حساسی که از OSINT جمع‌آوری می‌شوند، باید با دقت و احترام به اصول اخلاقی مورد استفاده قرار گیرند. تشخیص بین استفاده مشروع و ناپسند از اطلاعات OSINT امری ضروری است تا حقوق افراد و سازمان‌ها محترمانه حفظ شود.

ابزارها و تکنیک‌های پرکاربرد

در این بخش تعدادی ابزار و وبسایت کاربردی در راستای OSINT می‌بینیم که می‌توانند به ما در پروژه‌های OSINT کمک کنند.

- **osintframework.com:** این وب‌سایت تقریباً تمامی ابزارهای مورد نیاز ما در هر زیرشاخه‌ای از OSINT را به صورت طبقه‌بندی شده در اختیار ما قرار می‌دهد.
- **geohints.com:** این وب‌سایت به شکل خارق‌العاده‌ای کشورها را بر اساس معیارهای متنوع دسته‌بندی کرده که در OSINT شدیداً مورد استفاده قرار می‌گیرد. به عنوان مثال در یکی از بخش‌ها، کشورها را برحسب شکل چراغ راهنمایی‌ورانندگی طبقه‌بندی کرده است.
- **shodan.io:** شودان (Shodan) یک موتور جستجو است که به کاربر اجازه می دهد انواع خاصی از رایانه های (وب کم ها، روترها، سرورها و غیره) متصل به اینترنت را با استفاده از فیلترهای مختلف پیدا کند. این موتور جستجو برای متخصصان امنیت و نفوذگران بسیار مفید می‌باشد.فیلترهای اصلی شودان عبارتند از: شهر، کشور، مختصات جغرافیایی، سرور، سیستم‌عامل، پورت و...

اگر قصد تمرین و حل چالش‌های OSINT دارید، می‌توانید از لینک‌های زیر استفاده کنید:

- **medium.com:** اصلی‌ترین منبع برای تمرین و یادگیری OSINT، خواندن Writeupها و Storyهای دیگران در شبکه اجتماعی medium است. روزانه تعداد زیادی Story از پروژه‌های OSINT از متخصصان این حوزه روی medium قرار می‌گیرد که خواندن و بررسی آن‌ها برای علاقه‌مندان به OSINT بسیار مفید است.
- وبلاگ **osintteam.blog** در شبکه اجتماعی medium یکی از بهترین وبلاگ‌ها برای انتشار Writeupها در این حوزه است، خواندن Story پروژه‌های موجود در این وبلاگ را به شما پیشنهاد می‌کنم.
- **twitter.com/Quiztime:** این صفحه‌ی توییتر(x) روزانه تعدادی چالش OSINT توییت می‌کند که خواندن و بررسی آن‌ها می‌تواند به ما در تقویت مهارت‌هایمان کمک کند.
- **www.osintdojo.com:** این وب‌سایت علاوه بر آرشیوی از ابزارها و منابع برای جمع‌آوری اطلاعات، تعدادی چالش نیز دارد که شما می‌توانید آن‌ها را حل کنید و تواناهای خود را در OSINT تقویت کنید.
- **ctftime.org:** در مسابقات CTF غالباً سوالاتی از تیپ پروژه‌های کوچک OSINT مطرح می‌شود. شما می‌توانید سوالاتی با موضوع OSINT که در آرشیو سوال مسابقات CTF وجود دارد را به همراه Writeup آن در این وب‌سایت پیدا کنید.

نمونه‌های واقعی

مکان‌یابی تروریست‌های تونسی در رقه

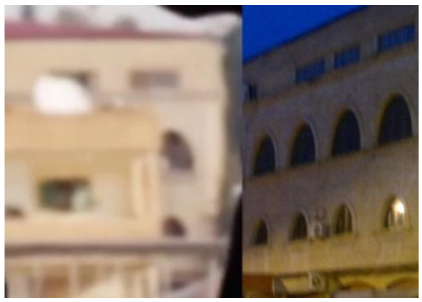
این هفته، یک فیلم ویدیویی منتشر شده که در آن جهادی‌های تونسی ادعا می‌کنند که آن‌ها مسئول قتل دو سیاستمدار سکولار تونسی در سال ۲۰۱۳ بوده‌اند. این فیلم به نظر می‌آید که در منطقه‌ای که تحت کنترل دولت اسلامی بوده است، تولید شده و از تکنیک‌های مختلف مکان‌یابی جغرافیایی برای تشخیص مکان فیلم‌برداری استفاده شده است. این ویدیو با دقت ویرایش شده و شامل سخنان مردی است که از زوایای مختلفی او فیلم گرفته شده است. در طی ۳۹ ثانیه از این ویدیو، یک پرچم و ساختمان‌هایی در پس‌زمینه دیده می‌شوند:

یکی از نکات جالب در مورد این ویدیو، وجود یک ساختمان جذاب با یک سقف قرمز است که در تصویر پشت پرچم قرار دارد. این ساختمان به طور واضح در ابتدای ویدیو قابل مشاهده است (حدود ۲۸ ثانیه از ابتدای ویدیو)، اما وقتی که چهار مرد در کنار آن نشسته و با دقت بررسی می‌شود، جزئیات بیشتری از این مکان آشکار می‌شود.

از نرم‌افزارهای ساده ویرایش تصاویر مانند Paint.net می‌توان برای ترکیب تصاویر این صحنه استفاده کرد تا همه‌ی ساختمان‌های موجود در پس‌زمینه به وضوح نمایش داده شوند و تصویری با نمای واضح‌تری از این ساختمان به‌دست آورد.

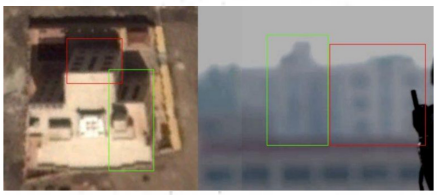
بنابراین ما باید به دنبال ساختمانی بلند با سقفی قرمز و پنجره‌هایی مستطیل‌شکل در طبقه‌ی بالا، و حداقل دو طبقه با پنجره‌های قوسی در پایین آن بگردیم.

ما با آغاز از شهر رقه، به بررسی نقشه‌هوایی شهرهای تحت کنترل دولت اسلامی پرداختیم. به سرعت یک ساختمان با سقف بزرگ قرمز را شناسایی کردیم، که به عنوان دانشکده مهندسی عمران شناخته می‌شد، و سپس جستجوی تصاویر پانوراما از این مکان را آغاز کردیم. این اقدام به ما اجازه داد تصاویر پانوراما از این محل را پیدا کرده و آن‌ها را با ساختمانی که در ویدیو دیده بودیم، مقایسه کنیم.



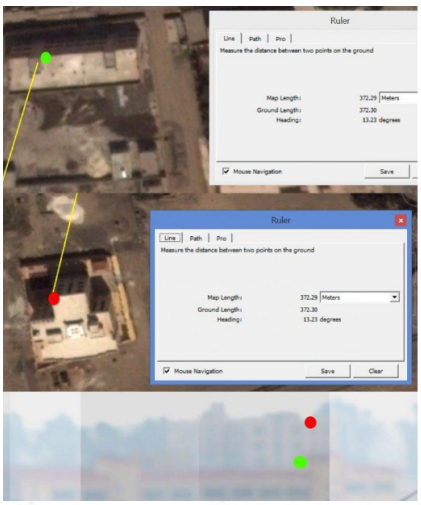
در ۳۰۰ متری غرب این ساختمان، میدانی عمومی با تیرک پرچمی بزرگ قابل رؤیت است، سایه‌ی ایجاد شده توسط این تیرک و خود پرچم در تصویر نقشه‌ی ماهواره‌ایِ زیر دیده می‌شود.

در تصویر ترکیبی، می‌توانیم در جنوب غربی ساختمان بالایی یک ساختمان دیگر را مشاهده کنیم. وجود عناصر مشخصی در دو تصویر به ما اجازه می‌دهد تا انطباق‌های واضحی را تشخیص دهیم، از جمله مکان قرارگیری پنجره‌ها در جلوی ساختمان و نیز ساختار بالایی آن ساختمان.



درختانی نیز در پشت ساختمان مشاهده می‌شوند، و تمام این عناصر نشان دهنده انطباق بسیار قوی این منطقه با تصاویر ویدیویی هستند که ما دیده‌ایم.

همچنین، می‌توان از بالای تصویر به تخمین موقعیت دقیق دوربین پرداخت. زیرا در تصویر، بخش‌های مشخصی از دو ساختمان را در یک خط مستقیم مشاهده می‌کنیم، این امر امکان خط کشی بین این دو نقطه و پیدا کردن مکان تخمینی دوربین را ممکن می‌سازد. در تصویر پایین، یک خط به وسیله علامت‌گذاری با رنگ‌های قرمز و سبز بین مناطق خاصی از ساختمان‌ها کشیده شده است که با موقعیت مشاهده شده در تصویر نقشه‌ی ماهواره‌ای همخوانی دارد.



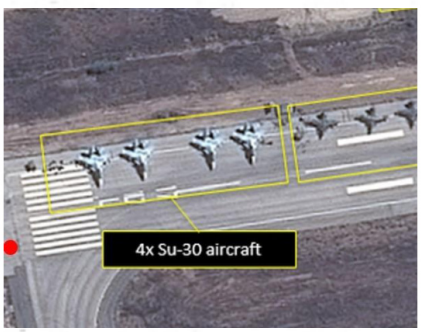
تعیین محل هواپیمای روسی در سوریه

در چند هفته اخیر، جابجایی نیروهای نظامی روسی به سوی سوریه و تصاویر منتشر شده توسط تحلیل‌های AllSource از فرودگاه بین‌المللی الاسد که نشان‌دهنده ورود هواپیماهای نظامی روسی به این فرودگاه است، توجه بسیاری را جلب کرد.

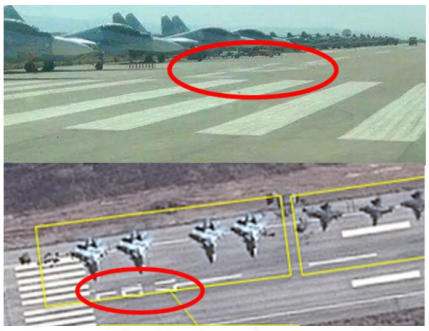
پس از آن عکسی در توییتر منتشر شد که ادعا می‌کرد هواپیماهای بالا را نشان می‌دهد.

حال آیا می‌توان ثابت کرد که این عکس در فرودگاه بین‌المللی الاسد گرفته شده است یا خیر؟ این پرسش را می‌توان به سادگی از طریق Google Earth و تصویر ماهواره‌ای اصلی که توسط تحلیل‌های AllSource گرفته شده است، پاسخ داد.

نخستین سوالی که باید پرسید، این است که دوربین در کجا قرار دارد؟ از عکس مشخص است که هواپیما در سمت چپ تصویر قرار دارد و نوارهای سفید دقیقاً جلوی دوربین قرار دارند. از طریق مقایسه با تصویر ماهواره‌ای، می‌توان به‌سادگی مکان دقیق دوربین را در سمت چپ تصویر تایید کرد، که با رنگ قرمز در تصویر مشخص شده است.



درنتیجه، می‌توانیم چهار هواپیمای ۳۰M-Su که در تصویر ماهواره‌ای مشاهده می‌شوند و به دوربین نزدیک‌تر هستند، شناسایی کنیم. این هواپیماها با دیگر هواپیماهای مشاهده‌شده در همان تصویر هم‌خوانی دارند و شاید به طور کامل واضح نباشد. همان‌گونه که در زیر نشان داده شده است، در تصویر معین امکان مشاهده نشانه‌های تمایزی در جلوی هواپیماهای ۳۰M-Su نیز وجود دارد.



درنهایت، از تصویر زمینی در Google Earth استفاده کردیم تا بیابیم که زمین در این منطقه با تصویری که در عکس مشاهده می‌شود هماهنگی دارد یا خیر. این اطلاعات بخاطر وجود تپه‌ها و کوه‌های واضح در پس‌زمینه تصویر Google Earth به ما داده می‌شود.

با به‌کارگیری این چند تکنیک ساده مکان‌یابی جغرافیایی، می‌توان به تأیید رسید که این عکس، ورود اخیر نیروهای روسی به فرودگاه بین‌المللی الاسد را نشان می‌دهد.

البته در این پروژه نیازی به استفاده از مواردی از قبلی طول‌وعرض جغرافیایی، موقعیت قرارگیری نسبت به خط استوا و... نشد. در پروژه‌های دیگر OSINT از این موارد به صورت حرفه‌ای و عملی استفاده می‌شود.

منابع

- bellingcat.com
- medium.com
- osint.ir

XAI

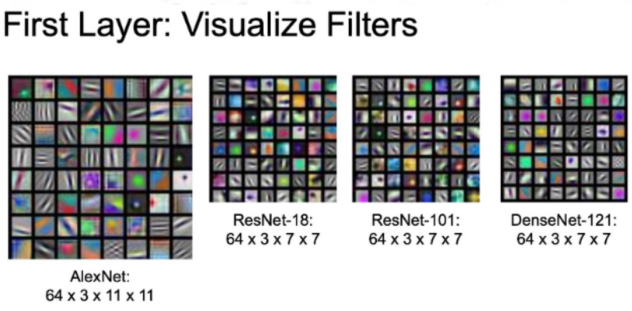
الگوریتم‌های هوش مصنوعی دقیقا بر چه اساسی کار می‌کنند؟ چقدر قابل اطمینان هستند و بر چه مبنایی تصمیم می‌گیرند؟

بهار دیبایی‌نیا

امروزه با گسترده شدن کاربرد شبکه‌های عصبی این سوالات بیشتر به وجود می‌آیند. درست است که ما در حوزه‌های بسیاری از مدل‌های یادگیری ماشین استفاده می‌کنیم ولی آیا در مسائل بسیار حساسی مانند تشخیص پزشکی و ... هم می‌توانیم به این مدل‌ها اطمینان کنیم؟ علت اصلی عدم اطمینان ما به این مدل‌ها علیرغم دقت بالایشان، ندانستن نحوه تصمیم‌گیری آن‌هاست.

ابزار هوش مصنوعی قابل توضیح (Explainable Artificial Intelligence–XAI) به ما در یافتن پاسخ این سوالات کمک می‌کنند. به نمونه‌های ساده‌ای از این موارد در شبکه‌های عصبی می‌پردازیم.

در شبکه‌های عصبی اگر وزن‌های لایه‌های اول را تصویرسازی کنیم، به اشکال ساده‌ای می‌رسیم که برای تشخیص ویژگی‌های سطح پایین تصویر مانند لبه‌ها، اشکال هندسی ساده و ... است ولی اگر لایه‌های عمیق‌تر مدل را تصویر کنیم به تفسیر خاصی از ورودی اولیه مدل نمی‌رسیم و فقط می‌توانیم ورودی آن لایه را تفسیر کنیم.

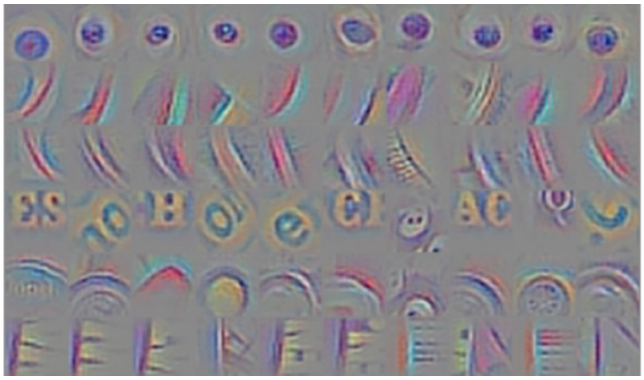


یک راه برای فهمیدن اینکه مدل به چه پیکسل‌هایی از ورودی حساس است و براساس آنها تصمیم می‌گیرد، به وسیله تغییراتی در ورودی صورت می‌گیرد به این صورت که در تصویر ورودی بلوک‌هایی قرار می‌دهیم و می‌بینیم که ایجاد آنها چه تاثیری روی خروجی شبکه می‌گذارد برای مثال در تصاویر پایین توزیع احتمالاتی خروجی آخر (احتمال هر دسته) در دسته‌بندی (classification) چگونه تغییر می‌کند.

یک روش دیگر در دسته‌بندی، گرفتن مشتق خروجی لایه آخر(امتیاز هر دسته (class score)) نسبت به پیکسل‌های تصویر ورودی می‌باشد. به زبان ساده می‌خواهیم بدانیم تغییرات کدام پیکسل‌ها در تصاویر ورودی تغییر بیشتری روی امتیاز هر دسته می‌گذارد.
از همین روش می‌توانیم برای فهمیدن اینکه هر نورون در لایه‌های عمیق نسبت به چه ورودی حساس است، استفاده کنیم با این تفاوت که بجای گرفتن مشتق خروجی لایه آخر، مشتق آن نورون را نسبت به پیکسل‌های ورودی می‌سنجیم.

در شکل پایین در هر ردیف ابتدا ورودی‌هایی را یافته که خروجی یک نورون به ازای آن ورودی‌ها زیاد است و سپس دامنه‌ای از تصویر ورودی که آن نورون دریافت می‌کند (receptive field) را جدا می‌کنیم.

سپس با استفاده از روشی که توضیح داده شد مشخص می‌کنیم که نسبت به کدام یک از پیکسل‌ها حساسیت بیشتری دارد.



به صورت کلی روش‌های تفسیر و توضیح پذیری همچنان محدودیت‌های بسیاری دارند و نیازمند توسعه بیشتر هستند و به همین دلیل از حوزه‌های پررونق این دوره به شمار می‌روند.


RANDOMIZED
SMOOTHING
AN INTRODUCTION TO CERTIFIED ML

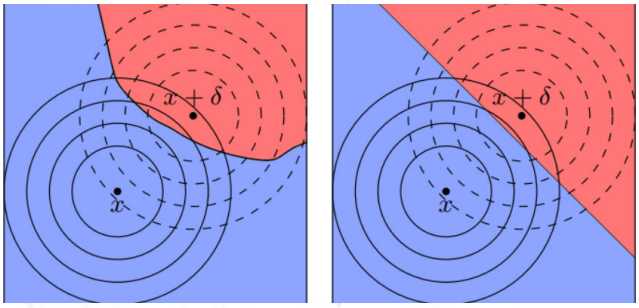
حسین گلی

RANDOMIZED SMOOTHING: AN INTRODUCTION TO CERTIFIED ML

یک مسئله جالب در یادگیری ماشین، یادگیری ماشین تضمین شده (Certified) است. این مسئله به ما می‌گوید یک شبکه عصبی که چیزی جز یک Function Approximator نیست، اگر به ازای ورودی x جواب درستی می‌دهد؛ آیا تضمینی برای درستی جواب به ازای $x+\delta$ برای دلتاهای کوچک می‌دهد یا خیر؟

در این متن قرار است کار آقای cohen را بررسی کنیم .می‌توانید به این لینک نیز برای بررسی دقیق‌تر مراجعه کنید.

از آنجایی که به ازای هر $f(x)$ دلخواه، ما نمی‌توانیم تئوری خاصی بدهیم، یک ایده جالب برای صحبت کردن به صورت کلی انجام می‌دهیم. فرض کنید به ازای هر تابع ورودی دلخواه شما نتیجه آن یا $f(x)$ را خروجی ندهید و بجای آن به ازای بینهایت نقطه $x+\delta$ کلاسی که بیشترین بار تکرار می شود (احتمال بالاتری دارد) را خروجی دهید. دقت کنید که این رویکرد به ازای هر تابع دلخواهی قابل انجام است.
درواقع ما یک تابع جدید به اسم $g(x)$ می سازیم به طوری که آن را می‌توان به صورت $g(x) = argmax_{c\epsilon Y} P(f(x+\varepsilon) = c)$ تعریف کرد که $\varepsilon \sim N(0,\sigma^2 I)$ می‌آید. در واقع ما تابع را نرم (smooth) کرده‌ایم . در تصویر زیر عملا این smooth کردن را می توانید مشاهده کنید.



تابع جدید ما خواص جالبی دارد. یک خاصیت جالب که cohen et. al آن را ثابت کرد این است که این تابع در یک شعاع، مقاوم (Robust) است؛ به این معنی که اگر در x نتیجه درستی بدهد، در $x+\delta$ هایی که $R > || \delta ||$ باشد نیز مقاوم است. شاید برایتان سوال باشد این مقدار R چه چیزی است؟ Cohen et. al اثبات کرده است که این شعاع برابر با:

$$R = \sigma^2(\Phi^{-1}(pA) - \Phi^{-1}(pB))$$

است که در این فرمول $p(A)$ و $p(B)$ احتمال دو محتمل‌ترین کلاس‌ها هستند و Φ^{-1} تابع توزیع تجمیعی معکوس توزیع نرمال است.
برای تخمین زدن احتمال‌های دو کلاس محتمل می‌توان از روش‌های monte carlo و نمونه برداری استفاده کرد. در عمل اضافه کردن نویز و آموزش دادن مدل ها با نویز، یک روش موثر برای ساختن مدل‌های robust نسبت به حمله‌ها به شبکه های عصبی هستند. (کافی است g را از روی f دلخواه بسازید و به صورت اثبات پذیر می دانیم g مقاوم است)

شماره اول (آبان‌ماه ۱۴۰۲)
سردبیر: امیرحسین رازلیقی
نایب دبیر: آرش یادگاری
مدیر مسئول: دکتر محمدحسین رهبان
صاحب امتیاز: انجمن علمی دانشکده‌ی مهندسی کامپیوتر
طراحان: معین آعلی، محیا ضرابی‌زاده و سجاد سلطانیان
نویسندگان (به ترتیب متن‌ها): امیرحسین رازلیقی - فاطمه‌حریرفروش - ایمان محمدی - عماد امام‌جمعه - امیرحسین حسن‌زاده - امیرمهدی نامجو - هیربد بهنام - معین آعلی - بهار دیبایی‌نیا - حسین گلی
ویراستاران علمی و ادبی: عرفان مجیبی - فرزوان ایرجی - امین رضانژاد - آرش فتانی فرشباف

بایت، اولین نشریه و جامعه علمی دانشکده‌ی مهندسی کامپیوتر دانشگاه صنعتی شریف است. بایت مکانی برای تبادل نظرات، بحث و آموزش به یکدیگر و آشنایی با مباحث تحقیقاتی و صنعتی در حوزه‌های علوم و مهندسی کامپیوتر است. تمامی مشتاقان به علوم کامپیوتر در هرجای دنیا، بی قید و شرط، می‌توانند عضو بایت باشند. اگر شما هم از آن دسته علاقمندان به علم هستید که یادگیری را در تبادل دانش می‌دانید، بایت جای درستی برای شماست!